ANL/NSE-21/42

Assessment of Fast Reactor Hot Channel Factor Calculation Capability in Griffin and NekRS

Nuclear Science and Engineering Division

About Argonne National Laboratory
Argonne is a U.S. Department of Energy laboratory managed by UChicago Argonne, LLC under contract DE-AC02-06CH11357. The Laboratory's main facility is outside Chicago, at 9700 South Cass Avenue, Argonne, Illinois 60439. For information about Argonne and its pioneering science and technology programs, see www.anl.gov.

Disclaimer

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor UChicago Argonne, LLC, nor any of their employees or officers, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of document authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof, Argonne National Laboratory, or UChicago Argonne, LLC.

Any opinions, findings, conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the Department of Energy Office of Nuclear Energy.

ANL/NSE-21/42

Assessment of Fast Reactor Hot Channel Factor Calculation Capability in Griffin and NekRS

prepared by
Emily Shemon, Yiqi Yu, Hansol Park, and Colin Brennan
Argonne National Laboratory

September 15, 2021

EXECUTIVE ABSTRACT

The DOE-NE Advanced Modeling and Simulation (NEAMS) program develops, demonstrates, and deploys predictive computational modeling tools for nuclear reactor design and analysis. Under the Fast Reactor Applications activity within the Application Drivers Technical Area of NEAMS, the ability to perform high fidelity "hot channel factor" (HCF) coupled multiphysics simulations has been identified as a valuable simulation capability.

HCFs are computed values that account for the impact on predicted peak fuel, cladding, and coolant temperatures due to uncertainties in the as-built reactor's material properties and geometry as well as uncertainties due to modeling approximations. Reduction in computed HCF values via reduction or elimination of modeling approximations may translate to significant economic savings if the reactor power can be raised due to the extra temperature margin gained. While limited historical datasets exist for sodium-cooled fast reactors (SFR), there are no available HCF data for lead-cooled fast reactors (LFR) outside of work generated previously within NEAMS.

Due to the significant advantages for multi-physics coupling offered by the MOOSE framework and accompanying MultiApp system, the MOOSE-based Griffin reactor physics code and MOOSE-wrapped NekRS thermal fluids code are desirable candidates for the high fidelity multi-physics modeling needed in advanced HCF simulations. Griffin and NekRS have been *individually* assessed for heterogeneously modeling the nominal condition of a reference LFR assembly design. Code coupling and workflow for perturbed calculations will be the focus of future work and are touched upon only briefly in this report.

Griffin was successfully used to model the 3D pin-by-pin heterogenous LFR assembly using multigroup cross sections generated by MC^2-3. Due to large memory requirements in Griffin for 33-group calculations, a 9-group cross section set was generated and studied to understand accuracy. Mesh and angular convergence studies were performed. Various solver options and acceleration schemes were tested, and the DFEM-SN solver with CMFD acceleration offered best performance and useability for this work. Significant calculational speedups in Griffin were observed from the start of the fiscal year to present which made this work feasible on intermediate computing clusters. Flexibility in material definition input, advances in postprocessing, and tracking power in fissionable/non fissionable regions were identified as needs and implemented throughout the year. Future work is recommended to reduce memory consumption for larger group calculations in Griffin as well as permit the user to normalize power including capture reactions.

NekRS, a GPU-oriented thermal-fluids simulation code based on the spectral element method (SEM), is developed at ANL, UIUC, and PSU. NekRS aims to leverage the present trend in GPU-based HPC systems to perform CFD on GPU-accelerated systems. NekRS was evaluated on several benchmark problems with flow properties similar to the LFR assembly. In summary, NekRS has the basic capabilities needed for HCF calculation. Some features in NekRS need to be improved to make it mature enough for HCF calculation, in particular ensuring accuracy and speed in RANS simulations.

ACKNOWLEDGEMENTS

This work was supported by the DOE-NE Advanced Modeling and Simulation Program. This report also includes material based upon work supported under an Integrated University Program Graduate Fellowship.

We gratefully acknowledge the computing resources provided on Bebop, a high-performance computing cluster operated by the Laboratory Computing Resource Center at Argonne National Laboratory.

The authors thank collaborators from Westinghouse Electric Company for providing reactor design information and guidance during previous phases of this work.

Guidance and support from Griffin developers (Changho Lee, Javier Ortensi, Yaqi Wang, Zachary Prince and Yeon Sang Jung) was very much appreciated.

Guidance and support from NekRS developers (Elia Merzari, Dillon Shaver, Ananias Tomboulides, Paul Fisher) was very much appreciated.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

1 Introduction

The U.S. Department of Energy, Office of Nuclear Energy Advanced Modeling and Simulation (NEAMS) [1] Campaign aims to develop, demonstrate, and deploy predictive computer methods for the analysis and design of nuclear reactor phenomena. The Application Drivers Technical Area within NEAMS is tasked with assessing the readiness of physics tools for specific reactor applications (fast reactor, microreactor, molten salt reactor, light water reactor, and others). By applying the codes to specific problems of interest to the nuclear community, user experience is built, code improvement feedback is generated, and sample working problem inputs are developed. This work assesses the ability of the reactor physics code Griffin [2] and thermal fluids code NekRS [3] to perform advanced hot channel factor (HCF) simulations for fast reactors.

Numerous uncertainties are involved in the predictions of reactor design parameters, including theoretical and experimental analysis uncertainties, instrumentation uncertainties, manufacturing tolerances, correlation uncertainties, and method and simulation uncertainties. These uncertainties impact the computed and actual peak cladding, fuel, and coolant temperatures in the system. The peak temperatures in the as-built system must nevertheless be maintained at safe margins away from maximum temperatures that could compromise the integrity and performance of the materials. The impact of uncertainties on the temperature predictions is typically accounted for through the assessment of hot channel factors, which consider the increase in reactor temperatures due to specific uncertainties. The goal of this work is to reduce or eliminate HCFs using high-fidelity multi-physics simulations that better account for the uncertainties associated with HCFs. If HCF values can be reduced through advanced modeling and simulation, the nominal peak cladding temperatures can be raised, resulting in higher power and economic gains.

In previous work [4-7], the NEAMS campaign employed PROTEUS [9] and Nek5000 [10] for high-fidelity sodium-cooled fast reactor (SFR) and lead-cooled fast reactor (LFR) HCF calculations using an offline, one-way coupled workflow. This resulted in the first known HCF dataset for LFRs [11]. The high-fidelity simulations reduced and/or eliminated geometrical approximations, and the computational scalability of these codes permitted advanced physics models including pin-by-pin heterogeneous transport, large eddy simulations (LES) and Reynolds Averaged Navier Stokes (RANS) for turbulence modeling, and conjugate heat transfer for computing heat transfer in the fuel and cladding zones. When compared to legacy HCF data from the EBRII sodium-cooled fast reactor [11], important conclusions were drawn. First, advanced modeling and simulation can reduce HCF in many cases. Second, HCF simulations should be performed for the design of interest rather than using a generic dataset based on reactor type. This is because geometric and material differences across designs may impact the resulting HCFs.

Building upon the success of PROTEUS and Nek5000, this work aims to leverage high fidelity tools which can be coupled through the Multiphysics Object Oriented Simulation Environment (MOOSE) framework [12] for more robust multiphysics coupling. The MOOSE-based Griffin reactor physics code and MOOSE-wrapped NekRS thermal fluids codes were chosen since they can be coupled via MOOSE's MultiApp system. Since both Griffin and NekRS are newly developed tools which are undergoing constant development and improvements, the two codes were assessed *individually* for performing their ability to solve the nominal condition calculation needed as the basis for all hot channel factor simulations. Studies were performed on the lead-cooled fast reactor design used in the previous work, since the absence of LFR hot channel factors in the literature creates an opportunity for advanced modeling and simulation to fill a critical need.

Furthermore, the LFR design features an interesting combination of features different from conventional SFRs including heavy metal coolant and annular fuel. The presence of the ducted assembly geometry in the LFR facilitates modeling since fluid flow is isolated within the assembly. While the work here is targeted at hot channel factor simulations, the assessment has broader applicability as users seek to demonstrate and advance both Griffin and NekRS capabilities on various high fidelity (heterogeneous pin-by-pin) fast reactor assembly geometries.

This document is laid out in 9 chapters. This section covered the background to the HCF work and the motivation of this work. Section 2 describes the general physics code requirements for performing advanced HCF simulations. Section 3 describes the reference LFR design and modeling assumptions.

Section 4 describes the input creation process for MC^2-3 [13] and Griffin, and it discusses cross section accuracy and code improvements implemented into Griffin to enable this work. Section 5 describes simulation results and performance of Griffin. Section 6 describes benchmarks performed to assess NekRS's capability.

Sections 7 and 8 describe early explorations of tools that can assist with performing the perturbed calculations needed in hot channel factor work, as well as description of the future coupling mechanisms. Section 9 presents the conclusions of this work and next steps.

2 Code Requirements for Fast Reactor Hot Channel Factor Modeling

This section summarizes the code requirements needed to determine hot channel factors for fast reactors with ducted assemblies (e.g. conventional SFR or this LFR design). A single 3D fuel assembly (with reflective boundary conditions) is modeled under nominal and multiple perturbed conditions. While the modeling of an isolated assembly neglects flux and power gradients that may occur due to assembly position in the core, it is sufficient for developing generalized hot channel factors that can be applied to specific assemblies in a whole core analysis. Given these geometry assumptions, specific neutronics and T/H code requirements are detailed in the following sections.

2.1 Neutronics Requirements

The neutronics solver must calculate and extract accurate pin powers and heating sources in a full 3D heterogeneous (pin by pin) assembly model with reflected boundary conditions. This is typically achieved by running neutron transport simulations with sufficiently refined energy, angle, and spatial discretization, either with a method of characteristics or finite element method. While pin power reconstruction methods could be used in conjunction with homogenized pin cell methods, the impact of these modeling approximations would need to be quantified since some of the HCF are focused on modeling small changes in the pin geometry (i.e. cladding thickness). Specific to the LFR design considered in this work, the annular fuel has a streaming region in the helium-filled gap which should be properly accounted for. Detailed mesh/angle convergence studies and performance results with Griffin are discussed in Section 5.

The input multigroup cross section dataset should be prepared carefully to preserve spatial heterogeneity effects; this becomes even more important as the number of energy groups is reduced to save computational expense and reduce memory requirements. Generation of multigroup cross sections is discussed in Section 4.2.

The heating source produced by the reactor physics code should preferably be differentiated in both fuel and non-fueled (cladding) regions so that the thermal fluids code can appropriately assign heat sources and perform heat transfer between solid regions. The output data (pin by pin axial power distribution, differentiated by fuel/non-fuel) should be available by post-processing file-based data or by coupling in-memory via MOOSE-based transfers. Output processing (which is specified in the input file) is discussed in Section 4.3.

Most of the studied hot channel factors require singular perturbations to the input which can easily be applied to the nominal condition input. However, the HCFs that involve compositional changes (e.g. fissile fuel content maldistribution) require significant input changes across simulations, and the ability to modify inputs should be facilitated as much as possible. The discretized geometry corresponding to pins should be accessed easily for assigning new materials, for example. Ability to specify microscopic cross sections and change the composition density is preferred. Discussion of updates to material input formats is discussed in Section 4.3.

2.2 Thermal Fluids Requirements

The thermal fluids code needs to take in detailed power distributions and perform pin-level heterogeneous 3D assembly calculations with/without wire wrap. A RANS model should be available for turbulence modeling in order to reduce the computational cost. HCF calculation also require the T/H code to have conjugate heat transfer module since a lot of uncertainties originate in the solid domain. The code should reproduce the solid temperature properly for HCF calculation.

The requirement also includes the flexibility of characterizing the fluid properties and power distribution.

2.3 Perturbation Requirements

The previous two sections indicate requirements for performing the nominal condition simulation which is the focus of this report. To obtain each HCF, perturbations are applied to the nominal condition simulation based on known uncertainties or statistical distributions. The use of a workflow manager such as MOOSE's Stochastic Tools which automatically perturbs the input according to statistical distributions would be highly beneficial. A preliminary exploration of MOOSE's Stochastic Tools applicability to this problem of interest is given in Section 8.

2.4 Coupled Multiphysics Requirements

The normalized pin power distribution calculated by neutronics is typically not strongly influenced by small temperature changes in the coolant or fuel of a fast spectrum reactor. Therefore, a one-way, once-through coupling is typically used where a reasonable temperature guess is assumed in neutronics, neutronics calculates power and transfers to the thermal fluids solver which calculate the velocity and temperature distributions in both fluid and solid. While two-way coupling is not strictly needed, it would be beneficial for verifying the accuracy of the once-through coupling. Additionally, on-line (in memory) coupling would simplify the information transfer and permit two-way coupling where neutronics cross sections could be re-interpolated based on temperature updates. One issue that may arise with on-line coupling is the disparate computational resource requirements for neutronics and thermal fluid codes, which tend to be dominated by the latter. The option to run them separately on different resources may override the benefits of online coupling depending on the how often the neutronics solution can be re-used.

3 Lead-Cooled Fast Reactor Modeling Problem

This report focuses on assessing Griffin and NekRS on a reference lead-cooled fast reactor assembly design provided by Westinghouse Electric Company (WEC). The ducted assemblies of this design make the simulation requirements of the LFR similar to conventional SFR designs. Additionally, there are no known published HCF datasets for LFRs, making the Westinghouse LFR an ideal candidate for this analysis.

3.1 Westinghouse Lead-Cooled Fast Reactor Design

WEC, together with an international team, is developing its next generation high-capacity nuclear power plant (NPP) based on lead-cooled fast reactor (LFR) technology. WEC provided technical details of a candidate LFR design in late 2018 and later helped prioritize the hot channel factors to assess with PROTEUS and Nek5000. All design details contained within this report are openly available [14].

Key parameters of the WEC LFR design are listed in Table 3.1. This medium-size, simple, scalable, and passively safe plant harnesses a liquid lead-cooled, fast neutron spectrum core operating at high temperatures in a pool configuration reactor. The power output is 950 MWth (~450 MWe). The core map is provided in Figure 3.1. The core consists of three concentric fuel assembly zones, two banks of control assemblies, safety, reflector, and shield assemblies. In this work, hot channel factors are analyzed for the *inner zone* fuel assembly type only, a cross section of which is shown in Figure 3.2. The impact of neighboring assemblies is not accounted for in this analysis. The generated HCF are generic enough to be applied to specific pins or assemblies in a full core analysis to yield actual temperatures.

Table 3.1. Key parameters of Lead-Cooled Fast Reactor (LFR) model

Parameter	***Value***	***unit***
Thermal power	950	MW
Active core height	105	cm
Number of fuel pins per assembly	127	-
Gap thickness	0.175	mm
Duct thickness	3.5	mm
Fuel pin diameter	10.7	mm
Fuel pin pitch	13.3	mm
P/D	1.24	-
Cladding thickness	0.9	mm
Fuel pellet outer diameter	8.55	mm
Fuel pellet inner diameter	4.0	mm
Reynolds number	$5.1\text{x}10^4$	-
Prandtl number	0.0136	-
Peclet number	705.21	-

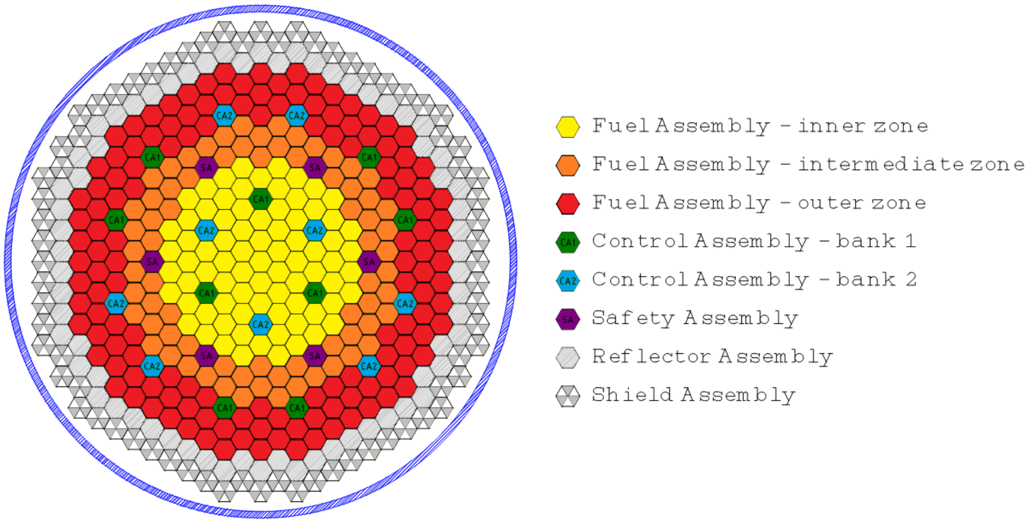

CA1
CA2
SA
Fuel Assembly - inner zone
Fuel Assembly - intermediate zone
Fuel Assembly - outer zone
Control Assembly - bank 1
Control Assembly - bank 2
Safety Assembly
Reflector Assembly
Shield Assembly

Figure 3.1. LFR core map

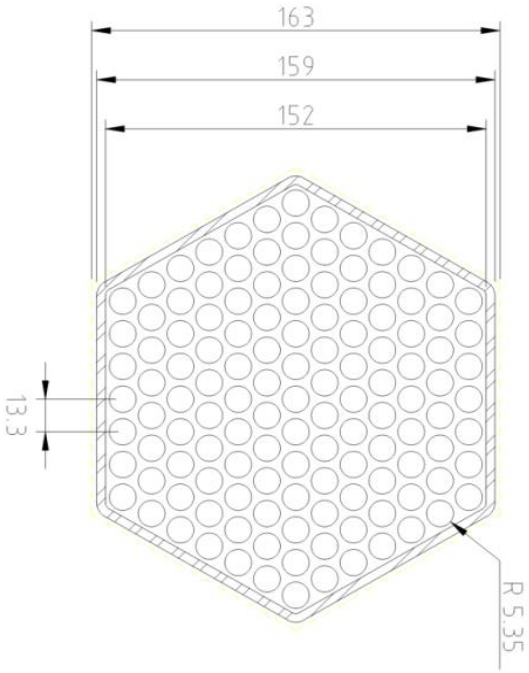


Figure 3.2. LFR fuel assembly geometry

The fuel assembly pitch is 16.3 cm with a 4 mm lead-filled gap between assemblies and 3.5 cm thick duct wall. Each assembly contains 127 cladded fuel pins arranged in a triangular lattice with pitch 1.33 cm within a hexagonal wrapper (duct). Each fuel pin has a cold fuel inner/outer diameter of 4.00/8.55 mm respectively, a fuel-cladding gap of 0.175 mm and a cladding outer diameter of

10.7 mm with a cladding thickness of 0.90 mm, depicted in Figure 3.3. Helium gas is present in the empty regions. To minimize the flow speed and consequently mitigate corrosion issues, a relatively wide lattice (P/D=1.24) design is adopted. Grid spacers are planned to maintain pin spacing, rather than the wire wrap used in conventional SFR designs. Annular MOX fuel (UPuO) is envisaged for countries with spent fuel management policies supporting reprocessing and reuse of plutonium. Annular fuel is very beneficial for fast reactors because of its availability for both high power and high burnup. Most of the annular pellets irradiated up to high burnup showed central-hole shrinkage due to deformation and restructuring during irradiation. This shrinkage has a great influence on power-to-melt, which is a main factor in deciding the maximum power in the fuel design [15].

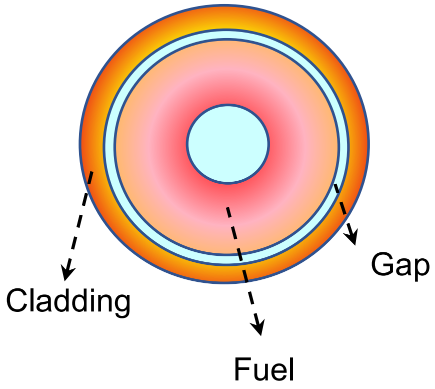


Figure 3.3. LFR annular fuel pin geometry.

The MOX (UPuO) fuel composition in the inner fuel zone is specified as a mixture of depleted uranium (0.25 wt% U-235) and enriched plutonium (68.769 wt% Pu-239, 1.759 wt% Pu-241), with 23% PuO_x / (PuO_x + UO_x). The cold fuel density is 10.423 g/cc. The wrapper (duct) and pin cladding are made of DS4 with density 7.970 g/cc. The lead coolant density is 10.402 g/cc. The axial regions of the fuel assembly are described in Table 3.2. The lower gas plenum through upper gas plenum are represented explicitly in neutronics (except for homogenizing the spring and tube), and regions above and below these are modeled homogeneously. Thermal expansion factors are applied to generate hot condition dimensions (1.021% fuel, 0.937% DS4 wrapper, 1.005% DS4 cladding, and 0.715% AISI316).

Table 3.2. Axial LFR zones and neutronics modeling information

Axial Zone	Neutronics Model Description	Hot Upper Z (cm)	Hot Height (cm)	Neutronics XS Temp. (K)
Lower core plate	Homogenized mixture AISI316, DS4, Pb	10.07	10.07	693
Inlet wrapper	Homogenized mixture DS4, Pb	40.86	30.79	693

Lower bundle grid/pins plug	Homogenized mixture DS4, Pb	47.42	6.56	693
Lower gas plenum tube	Heterogeneous DS4 wrapper & cladding, Pb Homogenized DS4/He tube	133.27	85.85	693
Lower thermal insulator	Heterogeneous DS4, Pb, He, YSZ	134.79	1.52	693
Active core	Heterogeneous DS4, Pb, UPuO, He, Pb	240.86	106.07	808 (Wrapper, Pb) 843 (Cladding) 1300 (He, UPuO)
Upper thermal insulator	Heterogeneous DS4, Pb, He, YSZ	242.37	1.52	923
Upper gas plenum spring	Heterogeneous DS4 wrapper & cladding, Pb Homogenized DS4/He spring	254.50	12.12	923
Upper bundle grid/pins plug	Homogenized mixture DS4, Pb	259.55	5.05	923
Outlet wrapper	Homogenized mixture DS4, Pb	353.42	93.87	923

3.2 Hot Channel Factors of Interest

While not used extensively in this present work, the list of priority HCFs for LFR are listed in the following table. These HCFs have been evaluated previously with PROTEUS and Nek5000 [8]. Future work will evaluate these with MOOSE-based coupling using Griffin and NekRS.

Table 3.3.List of priority LFR HCFs

LFR Hot Channel Factor	**Description**
Cladding thickness (subchannel flow area)	Assess impact of variances in cladding thickness due to manufacturing tolerance *Assumption: +/- 0.05 mm tolerance; change uniformly by maximum value in all pins due to meshing complexity*
Fissile fuel maldistribution	Assess impact of uncertainties in fissile content due to manufacturing tolerance *Assumption: +/-5% tolerance on Pu-239 enrichment; sample stochastically in all pins*
Coolant specific heat	Assess impact of uncertainties in lead coolant specific heat

	Assumption: +/-5% uncertainty in lead specific heat
Fuel thermal conductance across pellet-cladding gap	In low conductivity fuels with no bond, uncertainty exists for the thermal conductance across the pellet cladding gap. Estimate the sensitivity of the fuel temperature to the gap by performing analysis at two bounding cases: assuming perfect eccentricity of the fuel at (1) fresh fuel with open gap (nominal condition), and (2) burned fuel with closed gap. *Assumption: Detailed data on uncertainties in joint-oxide gain formation was not available, so the two bounding cases were analyzed. To assess the HCF, detailed fuel models and uncertainties are required which is outside the scope of the current SHARP toolkit and work scope.*
Coolant density	Assess impact of uncertainties in lead coolant density *Assumption: +/- 0.8% uncertainty in lead coolant density*
Coolant isotopics	Assess impact of uncertainties in lead coolant isotopics. *Assumption: Use "low" quality and "high" quality lead with varying Pb-208 contents to simulate lead sourced from different mines.*
Cladding conductivity	Assess impact of uncertainties in cladding conductivity. *Assumption: +/- 10% uncertainty in cladding conductivity*
Fuel conductivity	Assess impact of uncertainties in fuel conductivity. *Assumption:+/- 21.3% uncertainty in fuel conductivity*

4 MC^2-3 and Griffin Problem Setup

The major focus of this year's work was to assess and demonstrate Griffin's ability to model the nominal condition neutronics of the LFR heterogeneous assembly problem, assuming a fixed temperature profile. The desired output of this simulation is the detailed pin-by-pin axial power distribution in both the fuel and cladding zones. The nominal condition calculation involves explicit 3D annular pin geometry and therefore is computationally expensive. Modeling of the perturbed conditions needed for HCF will be performed in the future; these simulations naturally involve small tweaks of the nominal condition calculation inputs, so performance and accuracy of those should be comparable to the nominal condition calculation.

This section describes the reference MCNP model, efforts to generate appropriate input files for Griffin, a detailed investigation on the impact of multigroup generation methods, and improvements implemented in Griffin because of this project.

4.1 Monte Carlo Reference Model

A continuous energy MCNP 6.2.0 [16] reference solution was generated for the nominal condition to verify both the multigroup cross section set and the Griffin code accuracy. The same geometry was used in MCNP as Griffin: explicit heterogeneous geometry in the central five axial regions (lower/upper gas plena, lower/upper thermal insulator, and active fuel plane) and homogenized mixtures in the peripheral axial regions. The ENDF/B VII.0 neutron library was used to be consistent with MC^2-3-based cross sections in Griffin.

To ensure low statistical uncertainty, 100,000 particles/cycle, 100 inactive and 1000 active cycles were used. Statistical uncertainties on eigenvalue and axial power distribution in 20 sub-planes of the fuel plane were 5 pcm and less than 0.1%. For simplicity, the material temperatures used in MCNP were the nearest ones available to the nominal values as summarized in Table 4.1. In the active fuel region, the 808 K lead and duct and 843 K cladding were assigned as 900 K in MCNP. The 1300 K fuel was assigned to 1200 K in MCNP. The 693 K materials in planes below the active fuel were assigned to 600 K, and the 923 K materials in planes above the active fuel were assigned to 900 K in MCNP.

Table 4.1. Temperature approximations used in MCNP reference solution.

Axial Zone	**LFR Temperature (K)**	**Modeled Temperature in MCNP (K)**
Below Active Fuel	693	600
Active Fuel	808 (lead, duct) 843 (cladding) 1300 (fuel)	900 (lead, duct) 900 (cladding 1200 (fuel)
Above Active Fuel	923	900

The impact of using MCNP temperatures which differ from the actual LFR design was examined using MC^2-3 and turned out to be +36 pcm in reactivity and about 0.1% difference in axial power distribution. The power shape is the well-known cosine shape and almost identical for all pins in the fuel plane.

4.2 MC^2-3 Multigroup Cross Section Generation

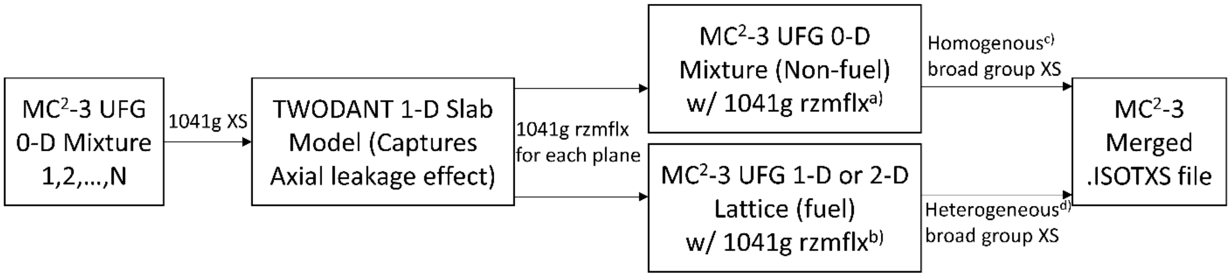
MC^2-3 multigroup cross sections were previously generated for PROTEUS calculations of the LFR assembly. The 33-group cross section data file (ISOTXS format) used in PROTEUS was converted to Griffin ISOXML [17] format. However, a memory issue was encountered when attempting to run the 33-group calculation with the version of Griffin at the time of investigation (git commit 9d535c0ff on 2021-07-01). Griffin required more than 1 TByte of memory and therefore exceeded the maximum memory available in the intermediate 80-core cluster being used. Therefore, the number of energy groups was reduced from 33 to 9 to reduce Griffin's memory consumption. Because a broader group structure was used, the accuracy of the 9-group cross section set was carefully investigated to determine whether 9-group cross sections could provide sufficient accuracy in the heterogeneous HCF simulations.

Two issues were checked: (1) accuracy of 9-group cross sections for the nominal case, and (2) whether the 9-group cross section set generated under the nominal condition is still applicable to perturbed problems. The model and calculational procedure in the MC^2-3 code is explained in Section 4.2.1 in order to understand how both the radial heterogeneity and the axial leakage effect are considered. Then, the second issue is discussed in Section 4.2.2.

4.2.1 Model and Calculational Procedure in MC^2-3

MC^2-3 has been widely used to generate subassembly-homogenized cross sections for fast reactor designs, coupled with the TWODANT S_N transport code to consider the core-wide spectral transition effect. Core nodal calculations are then typically performed using downstream codes such as VARIANT or DIF3D. In this application, the downstream code is a transport code for heterogeneous geometry (e.g., Griffin DFEM-SN, PROTEUS-MOC). In this work, a mixed representation of heterogeneous and homogeneous models was used in the 3-D assembly problem; fuel regions in the active core zone, lower and upper gas plena and thermal insulators are represented without geometrical approximation and inlet and outlet wrappers and lower and upper bundle grids & pins plugs are homogenized.

The procedure to generate cross sections for all regions of the problem is depicted in Figure 4.1. To perform the TWODANT calculation, 1041-group macroscopic cross sections for each plane were calculated by solving 0-D mixtures of each plane in MC^2-3 and used in the TWODANT 1-D slab calculation which emulates the axial heterogeneity of the assembly. The resulting 1041-group fluxes (contained in the rzmflx file) are used in the second MC^2-3 calculations for non-fueled axial zones (termed planes here) and the fuel plane. For each non-fueled plane, a 0-D mixture problem was solved, and the resulting UFG (2082-group) MC^2-3 flux solution was modified as Equation a) in Figure 4.1 using the rzmflx of the corresponding plane and used as a weighting spectrum for condensation as Equation c) in Figure 4.1. Note that even though planes for lower and upper gas plena and thermal insulators are represented with explicit geometry in Griffin calculations, homogenized broad group cross sections were used for each radial region under the assumption that radial heterogeneity (spatial self-shielding across the radial direction) is weak for non-fueled planes.


MC²-3 UFG 0-D Mixture 1,2,…,N
1041g XS
TWODANT 1-D Slab Model (Captures Axial leakage effect)
1041g rzmflx for each plane
MC²-3 UFG 0-D Mixture (Non-fuel) w/ 1041g rzmflx a)
MC²-3 UFG 1-D or 2-D Lattice (fuel) w/ 1041g rzmflx b)
Homogenous c) broad group XS
Heterogeneous d) broad group XS
MC²-3 Merged .ISOTXS file

a) $\tilde{\bar{\phi}}_g^{MC^2-3} = \bar{\phi}_g^{MC^2-3} \dfrac{\bar{\phi}_{G_T \ni g}^{TWODANT}}{\sum_{g \subset G_T} \bar{\phi}_g^{MC^2-3}}$ b) $\tilde{\phi}_{g,k}^{MC^2-3} = \phi_{g,k}^{MC^2-3} \dfrac{\bar{\phi}_{G \ni g}^{TWODANT}}{\sum_{g \subset G} \sum_{k'} \phi_{g,k'}^{MC^2-3} V_{k'} \Big/ \sum_{k'} V_{k'}}$ c) $\bar{\sigma}_{G,X}^i = \dfrac{\sum_{g \in G} \sigma_{g,X}^i \tilde{\bar{\phi}}_g}{\sum_{g \in G} \tilde{\bar{\phi}}_g}$ d) $\sigma_{G,X,K}^i = \dfrac{\sum_{g \in G} \sum_{k \in K} N_k^i \sigma_{g,X,k}^i \tilde{\phi}_{g,k} V_k}{\bar{N}_K^i V \tilde{\bar{\phi}}_{G,K}}$

g : UFG (2082G), G_T : TWODANT group (1041G), G : broad group, i : isotope (user-defined name), k : region, K : region that contains isotope i

Figure 4.1. Cross section generation procedure for a 3D heterogeneous assembly problem.

For region-wise cross sections in the fuel plane, a 1-D or 2-D problem shown in Figure 4.2 was solved using MC^2-3. The Collision Probability Method (CPM) was used for the 1-D model and the 2-D Method of Characteristics (MOC) was used for the 2-D model. The 1-D model was obtained from the conversion of the 2-D model by preserving volume fractions of each material. Region-wise flux solutions obtained by solving a UFG eigenvalue problem in such models considers the radial heterogeneity only, and thus, were adjusted by the TWODANT spectrum as Equation b) in Figure 4.1 to additionally consider the axial leakage effect. Then, region-wise cross sections were obtained by taking averaged values for each ring of fuel regions (7 rings), all cladding regions, all inner lead regions, the duct region, and the gap region using Equation d) in Figure 4.1.

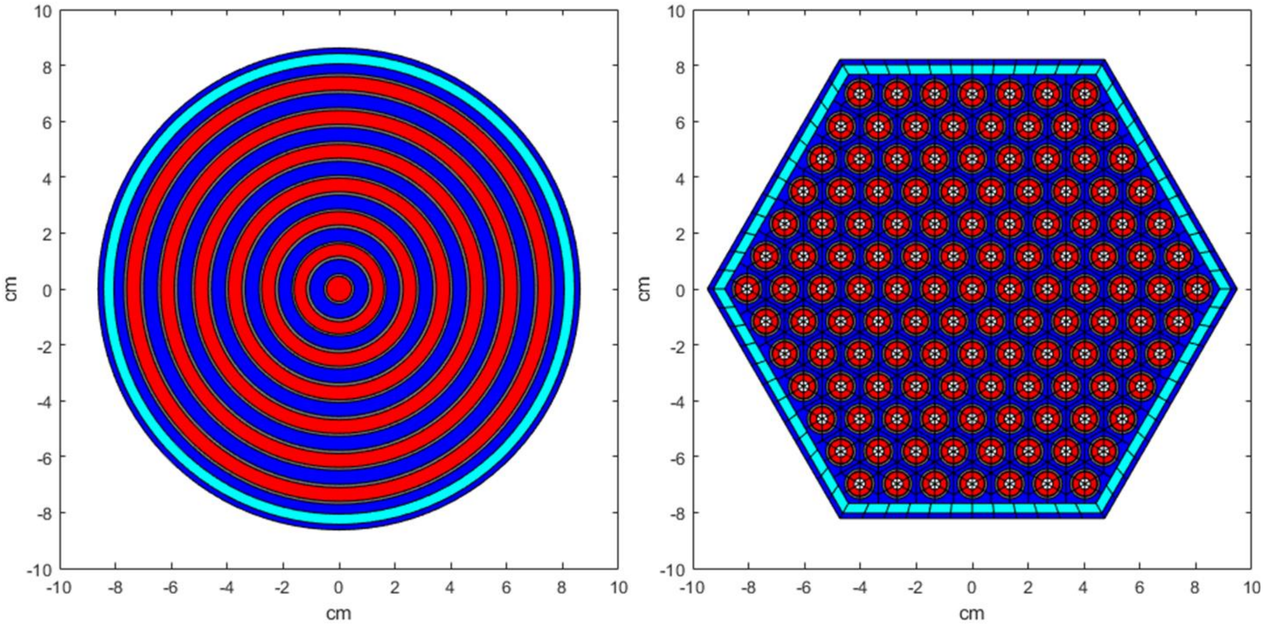


Figure 4.2. 1-D (left) and 2-D (right) model in MC^2-3 for cross sections in the fuel plane.

During this work, it was found that MC^2-3 had not been incorporating the TWODANT spectrum via Equation b) Figure 4.1 for generating region-wise cross sections (not subassembly-averaged cross sections) in 1-D or 2-D calculations, and thus, this was updated in the code locally, but this bug fix has not been pushed to the released version of MC^2-3 yet. This is especially important for 9-group calculations where the neglect of the axial leakage effect on within-group spectrum in the fuel region would cause non-negligible errors in 9-group cross sections due to coarse group widths of the 9-group structure. In this work, the conventional "ANL9" fast spectrum 9-group structure was used and no attempt at group structure optimization was made. For the user's benefit, relevant card options to generate region-wise cross sections with a TWODANT spectrum in 1-D or 2-D MC^2-3 calculations are summarized in Table 4.2.

Table 4.2. Essential Options in Input Deck of MC^2-3 to Generate Region-wise Cross Sections with TWODANT Spectrum in Heterogeneous MC^2-3 Calculations

Block	Card	Value
control	c_externalspectrum_ufg	Path of rzmflx
control1d	l_spatial_homogenization	F
material	i_externalspectrum(1)	Zone number of fuel plane in TWODANT calculation

4.2.2 Accuracy comparison of 33-group and 9-group cross sections in nominal case

The multigroup cross section generation procedure was verified against the MCNP continuous energy reference solution. *As the 33-group calculation was too memory intensive using Griffin on the available cluster, PROTEUS was used for the cross section verification*. The PROTEUS-MOCEX calculation used 20 axial meshes in the fuel plane and 48 total axial meshes, three annular subregions in the annular fuel pellet, L7T17 angular quadrature, and ray spacing of 0.01 cm. Multiple 33-group and 9-group cross section sets were generated with the 1-D and 2-D models of Figure 4.2. For each group structure and model, three different axial leakage and zoning assignments were tested:

(1) 1R/No Ax. Lkg: 1 axial cross section zone comprising all 20 axial meshes of the active fuel region without axial leakage effect (no use of TWODANT spectrum solution)

(2) 1R/Ax Lkg: 1 axial cross section zones comprising all 20 axial meshes of the active fuel region with axial leakage effect considered by using TWODANT spectrum solution via the MC^2-3 code update

(3) 3R/Ax Lkg: 3 axial cross section zones in fuel with axial leakage effect considered; this was done by using three axial cross section zones in the fuel region in TWODANT, using corresponding TWODANT spectra at three zones to generate cross sections and using consistent cross section zones in PROTEUS as well.

The first two cases tested the impact of excluding/including the axial leakage effect in the fuel zone. The third case tested inclusion of the axial leakage effect in the fuel zone but additionally used three axial cross section zones (zones for the 4 axial meshes at the top and bottom, and a zone for the 12 meshes in the middle).

Table 4.3 compares eigenvalue and axial power distribution results of PROTEUS and MCNP and Figure 4.3 shows axial power distribution errors. Since pin power errors were similar for all pins, only the center pin values are shown. The pin power here includes heat from fission and neutron capture in the fuel pellet only. Even though the heat from neutron capture is calculated in the cladding region, it was not counted in the comparison. The pin powers for both PROTEUS and MCNP were normalized in such a way that the average power in the axial fuel zone (20 axial meshes) is equal to one.

Table 4.3. Comparison of MCNP and PROTEUS Solutions

# of Groups	XS Model		K-eff	Δρ (pcm)	ΔPin Power (%) at Center Pin	
	Radial	Axial			RMS[a)]	Abs. Wgt.[b)]
MCNP			1.17169	±5	±0.08	-
33G	1-D	1R/No Ax. Lkg	1.17166	-2	0.93	0.65
		1R/Ax. Lkg	1.17285	+84	1.03	0.75
		3R/Ax. Lkg	1.17284	+84	1.11	0.82
9G		1R/No Ax. Lkg	1.16842	-239	0.39	0.26
		1R/Ax. Lkg	1.17223	+39	0.53	0.44
		3R/Ax. Lkg	1.17165	-3	0.96	0.84
33G	2-D	1R/No Ax. Lkg	1.17544	+272	0.74	0.52
		1R/Ax. Lkg	1.17555	+280	0.35	0.25
		3R/Ax. Lkg	1.17550	+277	0.36	0.28
9G		1R/No Ax. Lkg	1.17359	+138	1.80	1.24
		1R/Ax. Lkg	1.17600	+313	0.85	0.50
		3R/Ax. Lkg	1.17541	+270	0.41	0.26

a) $\sqrt{\sum_{k=1}^{N} e_k^2 \Big/ N}$ and b) $\sum_{k=1}^{N} |e_k| P_k^{MC} \Big/ \sum_{k=1}^{N} P_k^{MC}$, where $e_k = \frac{P_k^{PROTEUS} - P_k^{MC}}{P_k^{MC}}$ and N=20.

For 33-group calculation results, *eigenvalue* results of the 1-D cross section model agree well (within 84 pcm) with those of MCNP, while those of the 2-D cross section model have larger errors (280 pcm). On the other hand, the *pin power* distribution results of the 1-D cross section model have large deviations (higher power in the top and bottom and lower power in the middle), while those of the 2-D cross section model agree well with MCNP. An additional 2-D assembly problem (active fuel plane only) using the 2-D cross section model showed good agreement between PROTEUS-MOCEX and MCNP. Therefore, it is highly likely that overestimated eigenvalue results of the 2-D cross section model in Figure 4.2 are from the inconsistent use of assembly-homogenized cross sections in heterogeneous geometry for the four non-fueled planes, and the seemingly agreeable eigenvalue results of the 1-D cross section model are the consequence of error cancellation. The lower power in the middle planes for the 1-D cross section model shown in Figure 4.3 suggests larger leakage and thus is acting as lowering eigenvalue. The bottom left chart of Figure 4.3 shows that considering the axial leakage effect in the fuel zone (1R/Ax. Lkg) improves the result (reduces the error compared to MCNP). Using more axial cross section zones (3R/Ax. Lkg) has negligible impact on the result.

For 9-group calculation results, the improvement in the axial power distribution that results from including axial leakage effects (1R/Ax. Lkg) and then from adding additional cross section regions (3R/Ax.Lkg) are noticeable as shown in the bottom right figure of Figure 4.3. Using 9-group cross sections in which the axial leakage effect is not taken into account (1R/No Ax. Lkg blue line), the trend of higher power in the middle and lower power in the top and bottom is magnified. By considering the axial leakage effect and using more axial cross section zones, the axial power distribution was improved (blue to green to red) for the 2-D cross section model. For the 1-D cross section model, due to the existing bias of lower power in the middle and higher power in the top and bottom, the blue one looks the best due to error cancellation.

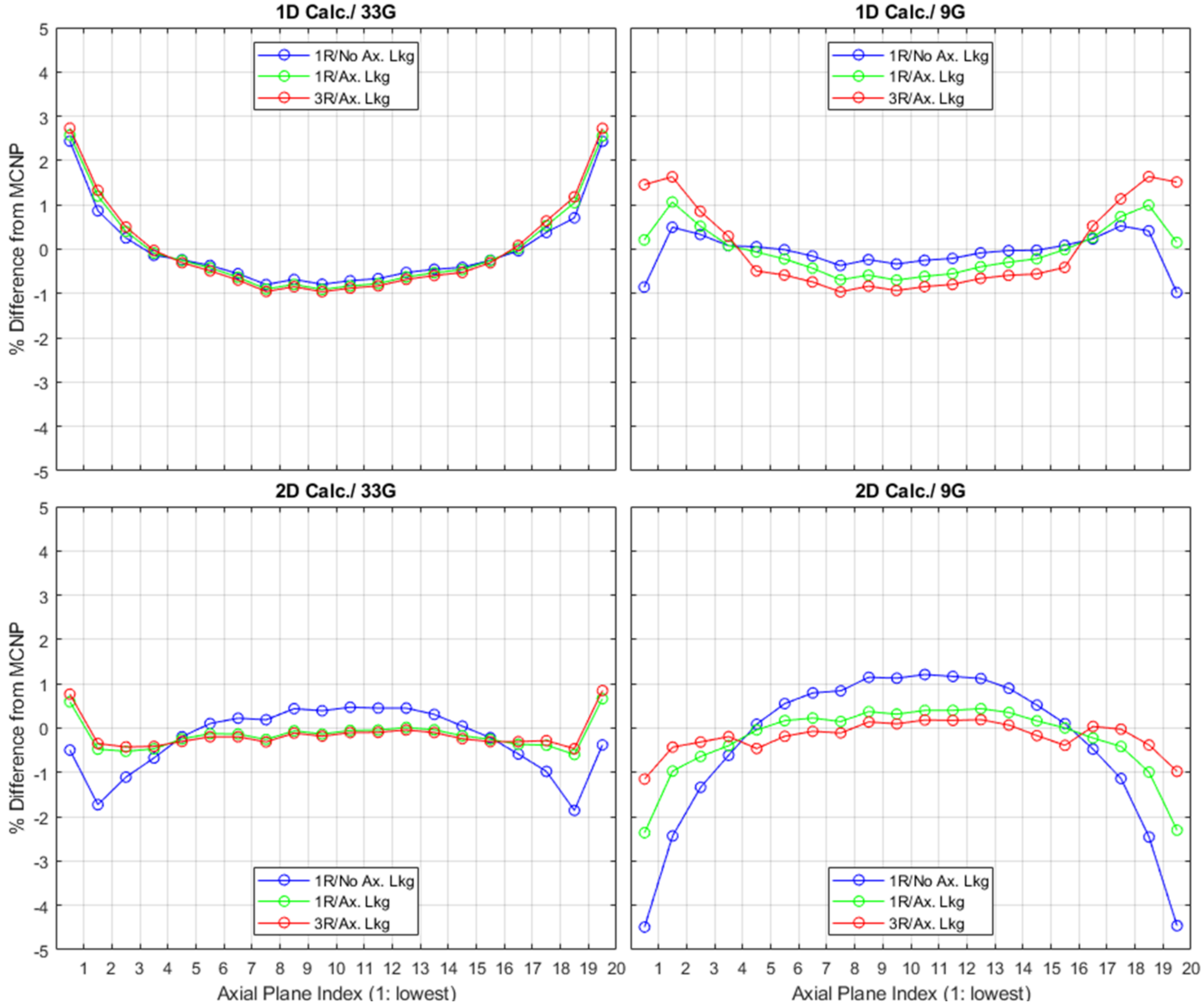


Figure 4.3. Relative errors of axial pin powers of PROTEUS at the center pin to those of MCNP.

These results indicate that 9-group cross sections give sufficiently accurate results so long as best practices were followed to account for spatial heterogeneity in the cross section generation process. The 2-D cross section model with the axial leakage effect and three axial cross section zones (3R/Ax Lkg) gives the best result in terms of power distribution with less than 1% error. The 1-D cross section model without the axial leakage effect and with one axial cross section zone (1R/No Ax Lkg) gives the best result owing to the error cancellation of the 1-D approximation of the radial geometry and the 9-group condensation without considering the axial leakage effect. *Given that we will be limiting Griffin calculations to 9-groups, we will use the "1R/Ax Lkg" cross section model that shows reasonable accuracy and includes the axial leakage effect*.

4.2.3 Investigation on validity of using 9-group cross sections for HCF evaluation

The previous section verified that 9-group cross sections can provide accurate power distributions for nominal conditions. This section investigates (1) whether these nominal 9-group cross sections can be applied for perturbed cases, and (2) the impact of small variations in power distributions on the temperature calculations and resulting HCF.

4.2.3.1 Impact on Power Distribution

9-group cross sections were verified to provide accurate power distributions in the previous section. This section considers whether the nominal 9-group cross sections can be applied to the perturbed 9-group state calculations or whether the perturbed state must be accounted for in the cross section generation process itself. For each perturbed case, 9-group and 33-group cross sections were generated under the exact perturbed condition (perturbed condition was accounted for in the cross section generation step). The perturbed problem was solved twice, once using exact perturbed cross sections and once using the nominal cross sections. The 9-group and 33-group calculations were then compared to see if the power difference induced by the perturbation followed the same trend regardless of group structure.

Five perturbed cases were selected: low and high lead quality, reduced lead density, and simplified ±6% change in plutonium fissile content. The fissile contents cases changed the Pu content uniformly for all pins by either +6% or -6% and the resulting power profiles are assumed to bound the more complex HCF case in which pins are individually perturbed.

Figure 4.4 to Figure 4.9 show axial power distributions of the nominal and perturbed cases (high lead quality, low lead quality, reduced lead density, +6% fissile content and -6% fissile content), obtained with 33-group and 9-group calculations. The dashed lines with "x" symbols lines depict the relative difference in power induced by the perturbation based on using nominal or perturbed cross sections.

For 33-group calculation results, all six perturbed cases show that using the nominal or perturbed cross sections result in the same final power solution in the perturbed case (the dashed lines with "x" symbols nearly match). In other words, the change in power distribution due to the perturbations can be accurately reproduced with the nominal cross section when performing a 33-group calculation. *On the contrary, when performing 9-group calculation, the choice of nominal or perturbed cross sections appears to impact the perturbed power solution (the dashed lines with "x" symbols do not match well in all cases)*. Since the magnitude of power change due to *physical* perturbations is often small, these small errors resulting from cross section methodology may be important in the thermal fluids calculation. As will be shown in Section 4.2.3.2, the cross section methodology induces uncertainties on the maximum temperatures in fuel, cladding, and coolant which may be larger than those introduced by physical perturbations. Thus, it is desirable to be able to perform more accurate 33-group calculations in the future even though 9-group cross sections were used for all Griffin calculations in this report.

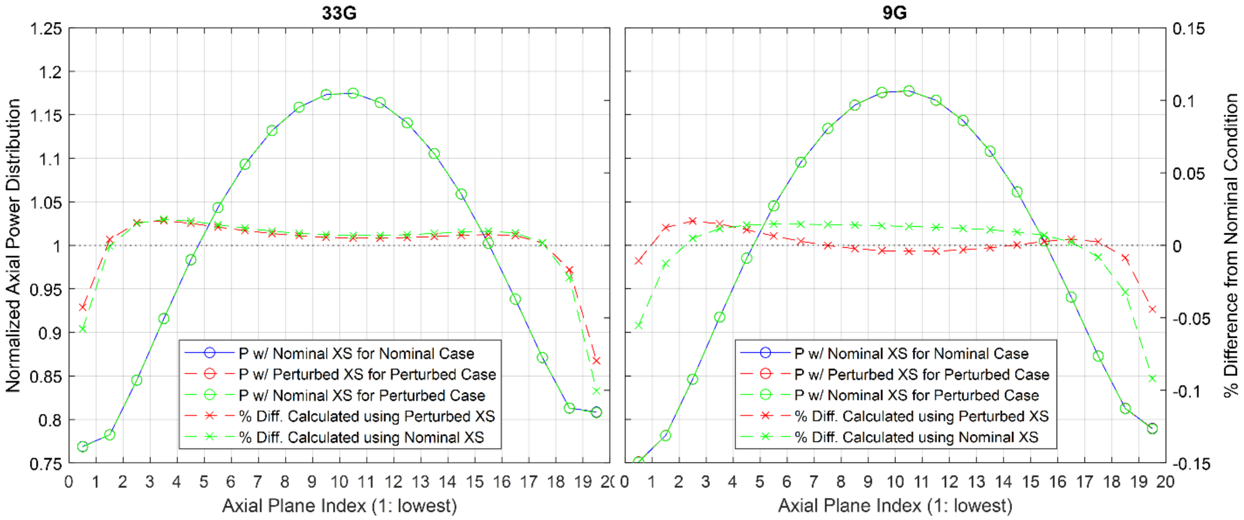


Figure 4.4. Axial power distributions of the nominal and perturbed case of high quality lead and their changes due to the perturbation obtained by 33G and 9G calculations.

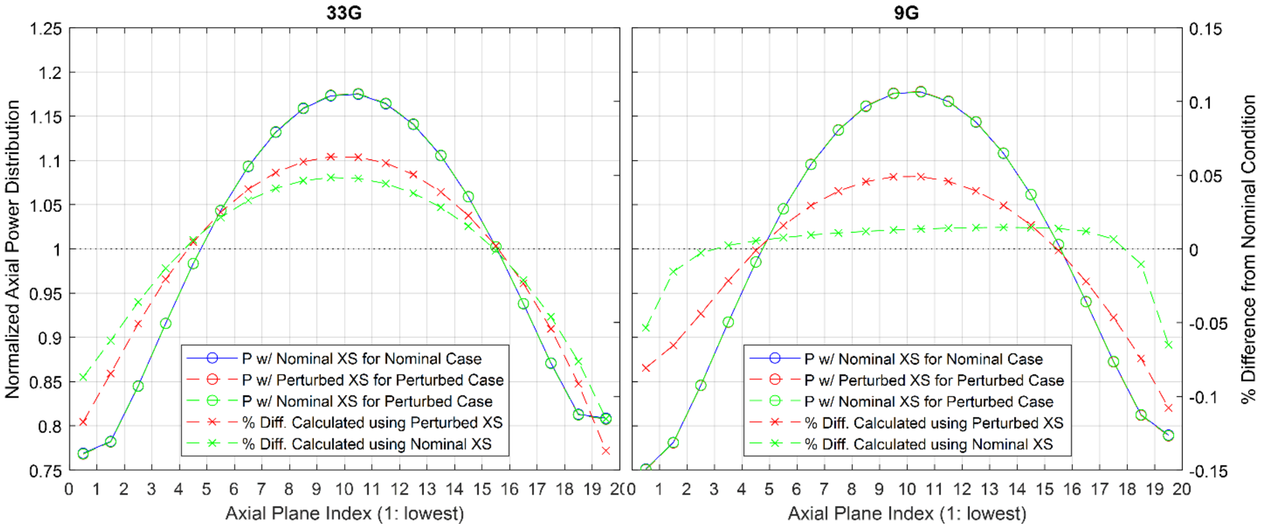


Figure 4.5. Axial power distributions of the nominal and perturbed case of low quality lead and their changes due to the perturbation obtained by 33G and 9G calculations.

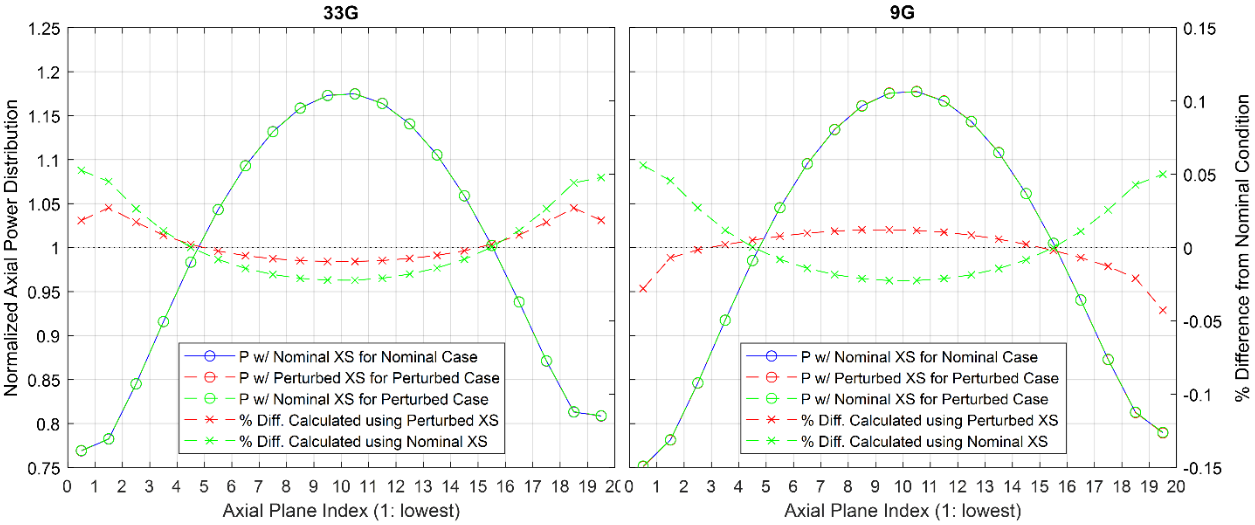


Figure 4.6. Axial power distributions of the nominal and perturbed case of reduced density lead and their changes due to the perturbation obtained by 33G and 9G calculations.

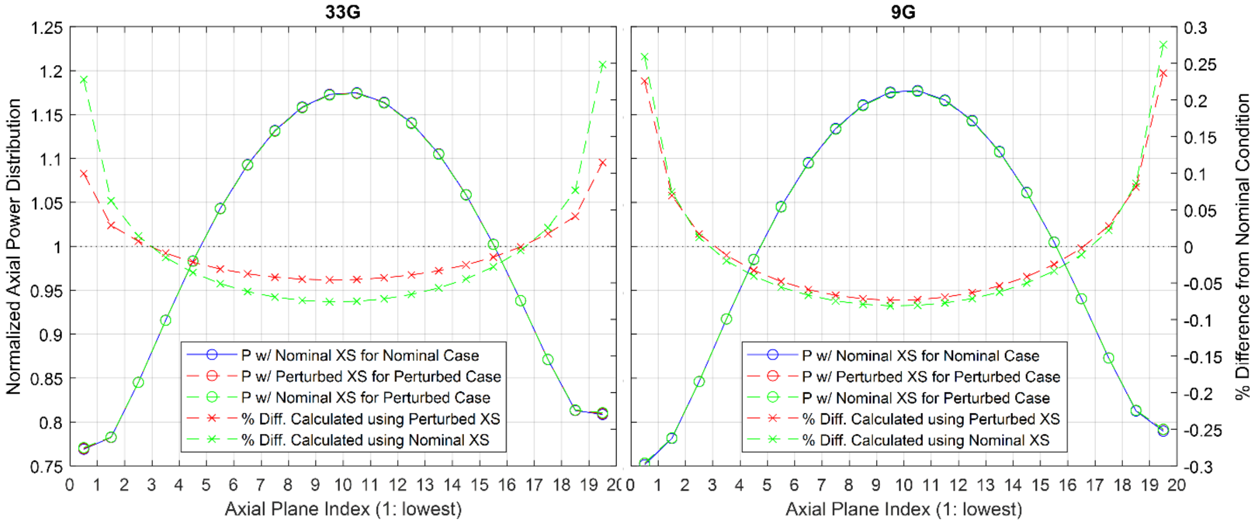


Figure 4.7. Axial power distributions of the nominal and perturbed case of uniform +6% fissile content and their changes due to the perturbation obtained by 33G and 9G calculations.

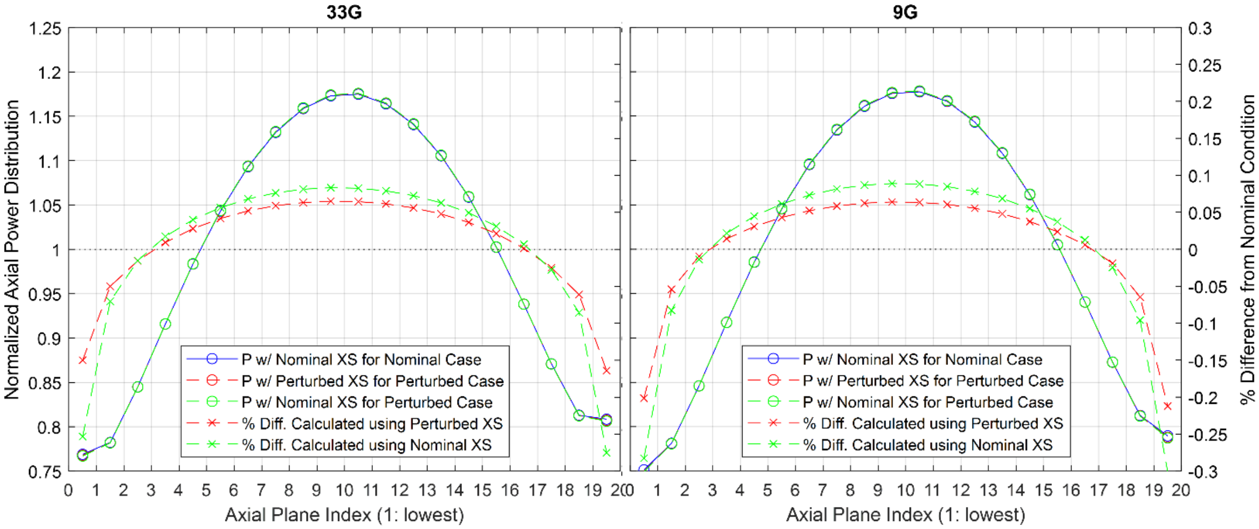


Figure 4.8. Axial power distributions of the nominal and perturbed case of uniform -6% fissile content and their changes due to the perturbation obtained by 33G and 9G calculations.

4.2.3.2 Impact on Temperature Distribution and HCF

As previously discussed, small differences were observed between multigroup power distributions and the MCNP continuous energy reference case, and we want to understand how these could impact the temperature calculation and hot channel factors. Multiple power distributions from MCNP, PROTEUS and Griffin were loaded into Nek5000 for coupled calculations. The purpose of the coupled calculation is to assess the impact of the cross section on the maximum temperature prediction. Table 4.4 lists the specific power distribution cases analyzed in Nek5000:

Table 4.4. Power profile cases considered for sensitivity study in Nek5000.

Case #	Case Name	MC^2-3 Options	Transport Case (P=PROTEUS G=Griffin)
0	MCNP CE	-	MCNP, nominal
1	P-33G-1D.Bug	33g-1D (includes impact of MC^2-3 bug for het. xs generation), nominal condition	P, nominal
2	P-33G-1D	33g-1D, nominal condition	P, nominal
3	P-33G-2D	33g-2D, nominal condition	P, nominal
4	P-9G-1D	9g-1D, nominal condition	P, nominal
5	P-9G-2D	9g-2D, nominal condition	P, nominal
6	G-9G-1D	9g-1D, nominal condition	G, nominal
7	Prt(LowQ.Lead)-P33G-1D-Nom.XS	33g-1D, nominal condition	P, perturbed

8	Prt(LowQ.Lead)-P9G-1D-Nom.XS	9g-1D, nominal condition	P, perturbed
9	Prt(LowQ.Lead)-P9G-1D-Prt.XS	9g-1D, perturbed condition	P, perturbed

Table 4.5 shows the peak temperature rise in fuel, cladding and coolant as impacted by different cross section methodologies. The discrepancy of temperature rise in coolant and cladding is smaller than 2^{o}C (2-3%) while the discrepancy of temperature rise in fuel is around 24^{o}C (<1%), which is reasonable based on the power discrepancy. It was observed that the cases are consistent in the prediction of the hot spot location.

Table 4.5. Impact of Cross Section Generation Method on Temperature (Nominal)

Case	**Fuel ΔT_{max} (Ratio to MCNP)**	**Cladding ΔT_{max} (Ratio to MCNP)**	**Coolant ΔT_{max} (Ratio)**
MCNP CE	1301.73 (ref)	29.18 (ref)	280.44
P-33G-1D.Bug	1312.89 (1.009)	28.76 (0.986)	280.44 (1.000)
P-33g-1D	1289.56 (0.991)	30.29 (1.038)	281.38 (1.003)
P-33g-2D	1303.20 (1.001)	29.75 (1.020)	281.02 (1.002)
P-9g-1D	1295.64 (0.995)	29.09 (0.997)	280.59 (1.001)
P-9g-2D (same as G-9G-1D)	1313.04 (1.009)	28.34 (0.971)	280.03 (0.999)

There are two potential scenarios for HCF calculations from a neutronics perspective:

1. **XS power error is much larger than the physical error power change**. This could be a problem since XS power error could dominate the TH solution.
2. **XS power error is smaller than the physical error power change**. This is not a problem since physical error power will dominate TH solution, as desired.

We tested a single case of low quality lead perturbation which is known to produce very small or negligible power perturbations from previous studies (testing the scenario where XS power may dominate the TH solution). As observed in Table 4.6, the impact of cross section power error resulted in a <1% impact on the cladding HCF: 0.996 (33g-1D), 0.998 (9g-1D), 1.000 (9g-1D-pert). This result cannot be generalized at this time as more case studies are needed.

Table 4.6. Impact of Cross Section Generation Method on Hot Channel Factor

Case	**Fuel ΔT_{max} (Ratio to MCNP)**	**Cladding ΔT_{max} (Ratio to MCNP)**	**Coolant ΔT_{max} (Ratio)**
P-33g-1D	1289.56 (ref)	30.29 (ref)	281.38 (ref)

Prt(LowQ.Lead)-P33G-1D-Nom.XS	1289.81 (1.000)	30.19 (0.996)	281.11 (0.999)
P-9g-1D	1295.64 (ref)	29.09 (ref)	280.59 (ref)
Prt(LowQ.Lead)-P9G-1D-Nom.XS	1295.45 (1.000)	29.04 (0.998)	280.47 (1.000)
Prt(LowQ.Lead)-P9G-1D-Prt.XS	1296.17 (1.000)	29.08 (1.000)	280.60 (1.000)

The power profiles from P-33g-2D (PROTEUS) and G-9g-1D (Griffin) were then used to calculate the HCF from cladding thermal conductivity (a purely thermal hydraulic HCF) to test the scenario where the physical perturbation should dominate any small power perturbations due to cross section errors. The results are listed in Table 4.7. Even though these two cases have slightly different power profiles and peak temperatures as a result of different cross section methodology, *the HCF are identical with the power profile from either PROTEUS or Griffin.* Since the same cross section cross sections are used in both nominal and perturbed calculations, it is likely that error cancellation occurs in the HCF (ratio of these two calculations) because both calculations have the same bias. It is likely that Griffin 9-group simulations will be sufficient for future HCF calculations, but more studies are needed.

Table 4.7. Cladding thermal conductivity HCF with power distribution from PROTEUS and Griffin

Code	**Nominal ΔT**	**Perturbed ΔT**	**HCF**
PROTEUS	28.6177	31.8587	1.113
Griffin	29.5807	32.9303	1.113

4.2.4 Summary of multigroup cross section findings

A procedure was developed and verified for generating accurate 9-group cross sections for the LFR assembly nominal conditions. MC^2-3 was updated to incorporate the axial leakage effect on 9-group cross sections for each region in a heterogeneous assembly calculation. Further eliminating approximations on the cross section geometry model by using the exact 2-D geometry instead of 1-D geometry and carrying three cross section zones in the active fuel plane reduced the power distribution error significantly. Meanwhile, 9-group cross sections generated using the 1-D geometry model with just one cross section zone in the active fuel plane gave similar accuracy in power distribution to those using the 2-D geometry model with three cross section zones as the result of error cancellation. If absolute temperatures are required for the calculation, 33g calculations are recommended. If only the HCF itself is important, the impact of cross section methodology is mitigated by error cancellation. Thus, 9-group cross sections generated using the 1-D geometry model of the "1R/Ax Lkg" case were used for the rest of this paper. In the future, 33g cross sections need to be enabled in Griffin to test accuracy of various energy group structures.

4.3 Griffin Input Preparation

The previous section described the generation of accurate multigroup cross sections for Griffin. This section explains the remaining Griffin input preparation for this work which is non-trivial due to the presence of numerous heterogeneous regions. Existing MC^2-3/PROTEUS inputs (cross section data file, mesh, material-to-mesh mapping) were leveraged for Griffin as much as possible using conversion utilities and python scripts (see Section 4.3.2). First, we discuss key improvements needed in Griffin to enable these calculations.

4.3.1 Improvements in Griffin

Because Griffin has not been employed routinely for pin-level heterogeneous simulations, some key improvements were needed immediately to meet the modeling criteria laid out in Section 2.1. These include:

- Performance improvement to make 3D heterogeneous assembly problem feasible on intermediate compute resources (implemented by Griffin team and not described here)
- Allow material definition based on microscopic cross sections and density, which permits easier modification of compositions in an input and ability to calculate region-dependent fission spectrum without excessive memory use in the code (implemented by Griffin team and described in 4.3.1.1)
- Tally axial power for each axial mesh within a block to extract axial pin power distribution, (implemented by Griffin team and described in 4.3.1.2)
- Easier designation of power tally region by material, (implemented by report authors and described in 4.3.1.2)
- Allow flux/power normalization based on both fission and capture reactions (implemented locally by report authors and described in 4.3.1.3)
- Tally energy deposition variable to yield heating in non-fuel regions (guidance given on how to do this by Griffin team, some scripts written to assist described in 4.3.2.2)
- Support hexagonal geometry meshing directly in MOOSE (implemented by MOOSE framework team at Argonne and not described here)

As noted, some of these improvements have already been implemented via coordination with Griffin and MOOSE framework developers. The new capabilities in Griffin which address these needs are described in Sections 4.3.1.1 through 4.3.1.3.

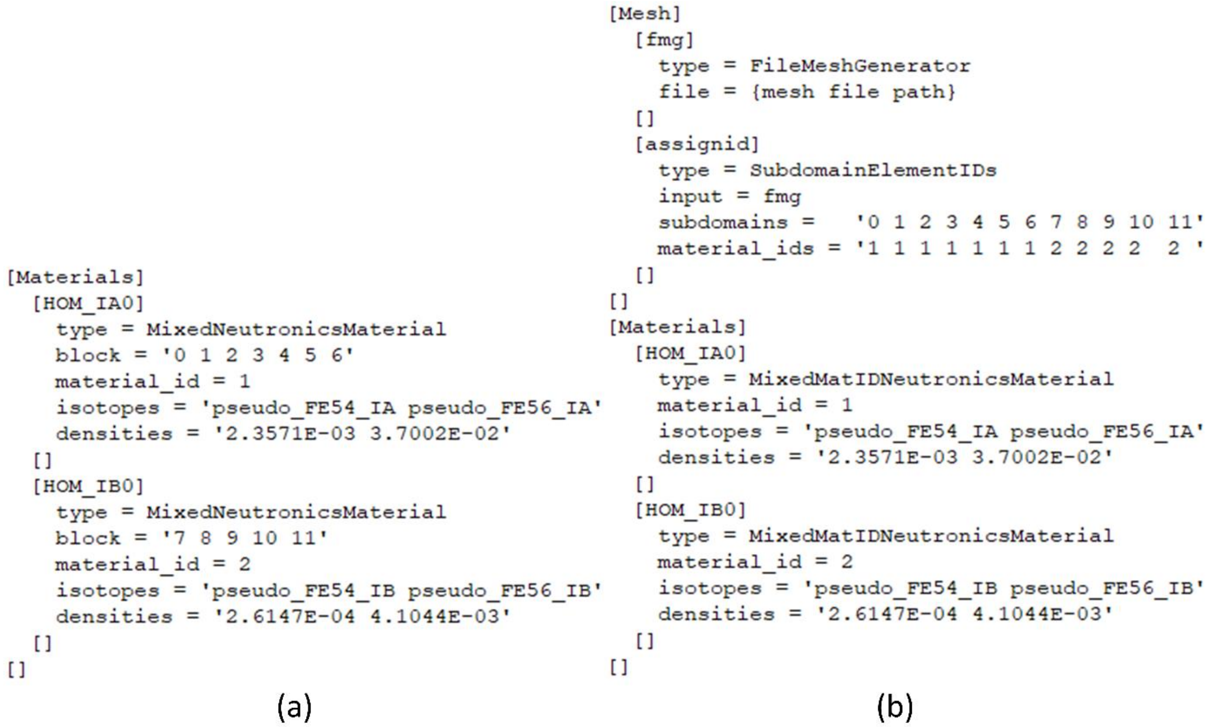
4.3.1.1 Addition of MicroNeutronicsMaterial

In fast spectrum reactor problems, the region-dependent fission spectrum, $\chi_g(r)$, should be prepared by *isotope-wise* fission source weighting. Microscopic cross sections need to be available for this weighting. This cannot be done using the macroscopic cross section approach, and neglecting this weighting has non-negligible impact on eigenvalue and power. To be specific, as isotopic fission source is not known a priori, composition-dependent fission spectrum is pre-generated with an assumed fission source weighting and used as a fixed value during the transport calculation. For example, the HMG4C method of generating composition-dependent fission spectrum [18] is to use a constant and uniform spectrum to compute isotopic fission sources. A simple test showed that this approximation can easily induce reactivity errors of about 300 pcm for fast spectrum problems.

Microscopic cross section approaches were already available in Griffin with the material types `MixedNeutronicsMaterial` or `MixedMatIDNeutronicsMaterial`. Both `MixedNeutronicsMaterial` and `MixedMatIDNeutronicsMaterial` specify a list of isotope names and atomic number densities from a specified ISOXML library. Meshes are mapped to materials within a Material input block for `MixedNeutronicsMaterial`, and materials are mapped to meshes within a Mesh input block for `MixedMatIDNeutronicsMaterial` as shown in Figure 4.9.

However, these material types are not the optimum choice due to memory and block-material mapping issues. For HCF calculations, the microscopic cross section library is generated only once and any isotopes in the library should be freely used in any material with an arbitrary isotope density. This is allowed in `MixedNeutronicsMaterial`, but to optimize the memory usage, the cross section library should be re-organized or manipulated for perturbed cases of different compositions. For `MixedMatIDNeutronicsMaterial`, the ISOXML file should be re-organized for a fixed set of compositions with fixed material ids for different problems. In either case, this is extremely onerous on the user particularly for HCF calculations in which small material perturbations are needed.

Here we describe the memory issues present with the existing material types. For `MixedNeutronicsMaterial`, material ids do not matter because mesh blocks are directly assigned to each material in a Material input block. However, if all the isotopes are defined under only one material id (one Multigroup_Cross_Section_Library in the YakXs format hierarchy in an ISOXML file) and isotopes are spread across several different materials, the Griffin code holds the data of all the isotopes, called the mixing table data, for each material in duplicate to calculate region-dependent fission spectrum using isotopic fission source weighting. This is a non-negligible waste of memory. To avoid this, an ISOXML file can be generated in such a way that each material id contains only its comprising isotopes in the file. This is not desirable in this work in which one ISOTXS file that contains microscopic cross sections of all isotopes is generated only once under the nominal condition and isotopic compositions can be freely changed in a Griffin input to solve any perturbed problems. That is, an ISOXML file may have to be re-generated or re-organized for different problems according to composition changes in each material id in an ISOXML file. Even ignoring the memory problem and using just one material id, specifying mesh block ids for each material makes an input unmanageable by users because users do not care about actual id values of mesh blocks in general. Similarly, the material-mesh mapping is established in a Mesh input block for `MixedMatIDNeutronicsMaterial`, which means that material ids should be different to distinguish materials. Thus, the difficulty of managing ISOXML files for every perturbed case stated above also occurs for `MixedMatIDNeutronicsMaterial`.



```
[Materials]
  [HOM_IA0]
    type = MixedNeutronicsMaterial
    block = '0 1 2 3 4 5 6'
    material_id = 1
    isotopes = 'pseudo_FE54_IA pseudo_FE56_IA'
    densities = '2.3571E-03 3.7002E-02'
  []
  [HOM_IB0]
    type = MixedNeutronicsMaterial
    block = '7 8 9 10 11'
    material_id = 2
    isotopes = 'pseudo_FE54_IB pseudo_FE56_IB'
    densities = '2.6147E-04 4.1044E-03'
  []
[]
```

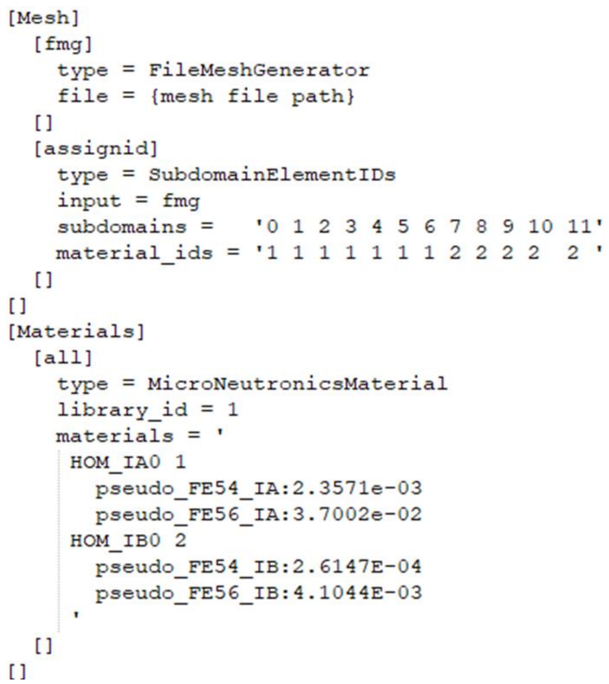
(a)

```
[Mesh]
  [fmg]
    type = FileMeshGenerator
    file = {mesh file path}
  []
  [assignid]
    type = SubdomainElementIDs
    input = fmg
    subdomains =   '0 1 2 3 4 5 6 7 8 9 10 11'
    material_ids = '1 1 1 1 1 1 1 2 2 2 2  2 '
  []
[]
[Materials]
  [HOM_IA0]
    type = MixedMatIDNeutronicsMaterial
    material_id = 1
    isotopes = 'pseudo_FE54_IA pseudo_FE56_IA'
    densities = '2.3571E-03 3.7002E-02'
  []
  [HOM_IB0]
    type = MixedMatIDNeutronicsMaterial
    material_id = 2
    isotopes = 'pseudo_FE54_IB pseudo_FE56_IB'
    densities = '2.6147E-04 4.1044E-03'
  []
[]
```

(b)

Figure 4.9. Sample input for (a) MixedNeutronicsMaterial and (b) MixedMatIDNeutronicsMaterial.

The Griffin development team addressed this issue properly by creating a new material type named `MicroNeutronicsMaterial`. *The biggest advantage of using* `MicroNeutronicsMaterial` *is that material ids used in an input are not tied with those in an ISOXML file*. As shown in Figure 4.10, one can freely assign any material ids independently from the data structure of an ISOXML file while putting all isotopes under the same Multigroup_Cross_Section_Library ID in the YakXs format hierarchy in an ISOXML file, which is now read as `library_id`, not `material_id.` Since only one material MOOSE object is created, only one mixing table data is stored in memory regardless of the number of material ids used. Additionally, since material ids are mapped to mesh ids in a Mesh input block, block (subdomain) ids do not need to be accessible or known by the user in the material block. Potentially, this material type can be incorporated to a new type of a MOOSE-native mesh generator that does not require the user to know any block ids.



```
[Mesh]
  [fmg]
    type = FileMeshGenerator
    file = {mesh file path}
  []
  [assignid]
    type = SubdomainElementIDs
    input = fmg
    subdomains =   '0 1 2 3 4 5 6 7 8 9 10 11'
    material_ids = '1 1 1 1 1 1 1 2 2 2 2  2 '
  []
[]
[Materials]
  [all]
    type = MicroNeutronicsMaterial
    library_id = 1
    materials = '
     HOM_IA0 1
       pseudo_FE54_IA:2.3571e-03
       pseudo_FE56_IA:3.7002e-02
     HOM_IB0 2
       pseudo_FE54_IB:2.6147E-04
       pseudo_FE56_IB:4.1044E-03
     '
  []
[]
```

Figure 4.10. Sample input for MicroNeutronicsMaterial.

Therefore, the new `MicroNeutronicsMaterial` option in Griffin allows (1) proper isotopic-dependent fission source weighting to determine the region dependent spectra, and (2) huge simplification for the user to define materials flexibly while maintaining lower memory consumption.

4.3.1.2 Update of HexagonalGridVariableIntegral

When tallying variables from hexagonal geometry in Griffin, a special type of VectorPostprocessor (VPP) named `HexagonalGridVariableIntegral` is typically used. Two parts of this VPP were improved in this work by the Griffin development team.

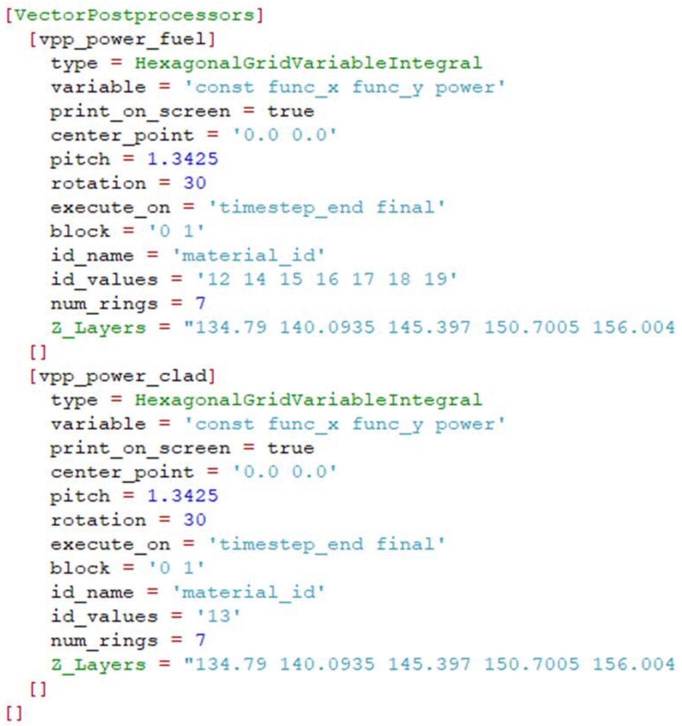
4.3.1.2.1 Z_Layers

Detailed axial power distributions output are required during HCF simulations. Previously, there was no way to tally variables within a block, meaning that axial mesh-wise powers could not be tallied unless an excessive number of axial blocks were defined, which causes performance issues in MOOSE-based applications. Since it would be very inefficient to add extra blocks only for tally purposes, *the VPP was updated to allow a mesh-wise tally within a block. One can specify a list of z coordinates in the ascending order that match boundaries in the z-direction*.

4.3.1.2.2 User-defined element extra integer id name for designating tally region

In the context of user-friendliness, the previous scheme of the VPP was not optimized because a user needs to know the block id values in which power needs to be tallied. These block ids are not immediately obvious especially in cases with heterogeneous geometry. In addition, blocks are often merged together to reduce the total number of blocks in the model, and once this is done there is no way of identifying specific elements within a merged block for tallying power. Thus, the block merging scheme was in effect restricted by where the user needed to tally power.

Since power is usually needed per material region, it makes more sense to use material ids to designate elements to tally power. *Generalizing this, the code was modified to permit user-specified extra element integer id names and values to identify elements on which to postprocess data*. The block merging scheme and the VPP input are no longer tied together, and users do not need to know actual block id values of elements of interest. Instead, the block input is used to restrict large portions of the geometry as the query on material id will only be performed on those blocks. Figure 4.11 shows a sample input for the VPP to tally powers at fuel and clad regions separately.

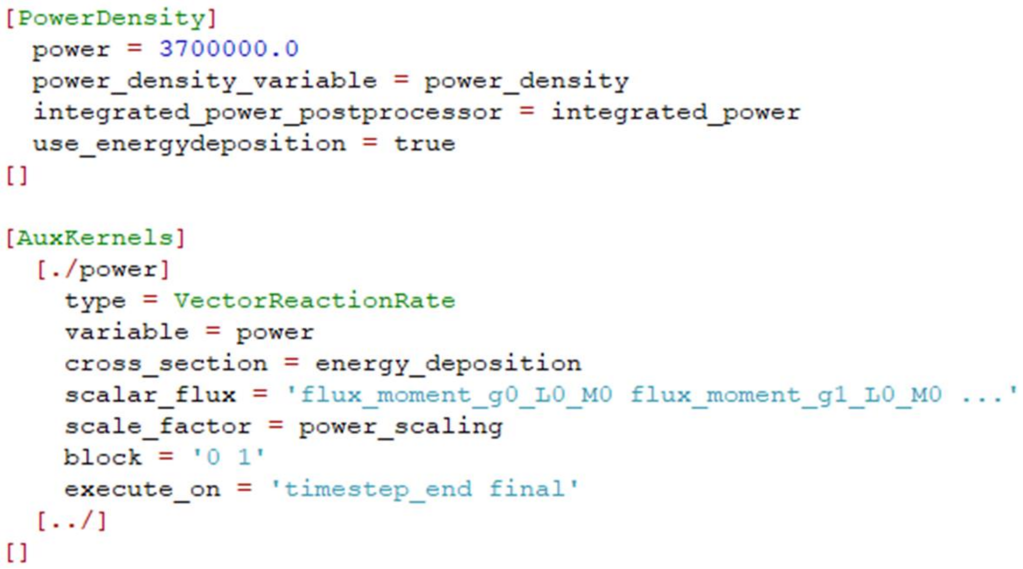


```
[VectorPostprocessors]
  [vpp_power_fuel]
    type = HexagonalGridVariableIntegral
    variable = 'const func_x func_y power'
    print_on_screen = true
    center_point = '0.0 0.0'
    pitch = 1.3425
    rotation = 30
    execute_on = 'timestep_end final'
    block = '0 1'
    id_name = 'material_id'
    id_values = '12 14 15 16 17 18 19'
    num_rings = 7
    Z_Layers = "134.79 140.0935 145.397 150.7005 156.004
  []
  [vpp_power_clad]
    type = HexagonalGridVariableIntegral
    variable = 'const func_x func_y power'
    print_on_screen = true
    center_point = '0.0 0.0'
    pitch = 1.3425
    rotation = 30
    execute_on = 'timestep_end final'
    block = '0 1'
    id_name = 'material_id'
    id_values = '13'
    num_rings = 7
    Z_Layers = "134.79 140.0935 145.397 150.7005 156.004
  []
[]
```

Figure 4.11. VPP sample inputs to tally powers at fuel and cladding regions.

4.3.1.3 Update of PowerDensity Action

The [`PowerDensity`] block is used to normalize flux inside the code to a user-specified total power,. Figure 4.12 shows the [`PowerDensity`] block in which total power of 3.7 MW is specified. Since the existing feature uses just kappa fission cross section to calculate power, additional energy deposition from neutron capture reactions is ignored. Thus, in this work, the code was modified locally to use the EnergyDeposition variable by adding the "`use_energydeposition`" flag as an additional input card. *This has not been merged into the main development branch of Griffin for a code management reason, and we request this functionality in Griffin sometime in the future*. By specifying the [`PowerDensity`] block in the input, flux is normalized to have the user-specified total power and the scaling factor is stored in the "`power_scaling`" variable by default, which in turn should be used in the AuxKernel that defines how the AuxVariable named "`power`" is calculated. In this way, the "`power`" AuxVariable becomes a properly normalized quantity to be used in VectorPostProcessor shown in Figure 4.11.



```
[PowerDensity]
  power = 3700000.0
  power_density_variable = power_density
  integrated_power_postprocessor = integrated_power
  use_energydeposition = true
[]

[AuxKernels]
  [./power]
    type = VectorReactionRate
    variable = power
    cross_section = energy_deposition
    scalar_flux = 'flux_moment_g0_L0_M0 flux_moment_g1_L0_M0 ...'
    scale_factor = power_scaling
    block = '0 1'
    execute_on = 'timestep_end final'
  [../]
[]
```

Figure 4.12. Flux normalization to a user-specified total power

4.3.2 Development of Python scripts to convert PROTEUS Input to Griffin Input

This section discusses the last few loose ends on creating Griffin input. An ongoing effort is taking place to develop a MOOSE-native meshing system for fast reactor cores design, but in the meantime, python scripts were developed to set up the Griffin input mesh and material mapping based on existing PROTEUS inputs. To generate the mesh and material blocks for Griffin, a user needs to provide the control input for the python script and following inputs of PROTEUS:

1. UFMESH file of the PROTEUS mesh tool (MT_MeshToMesh) for radial mesh information
2. Control input file of the PROTEUS mesh tool (MT_BuildAxialMesh) for axial mesh information
3. Assignment file of PROTEUS for material-mesh mapping
4. Microscopic ISOTXS file for cross section

The outputs of the script are `[Mesh]` and `[Materials]` blocks in the format of Figure 4.10 that can be directly used for a Griffin input by copy and paste and a microscopic ISOXML file. Secondary outputs are the material-to-block map and element type per merged block IDs that can be useful for users to build other parts of a Griffin input.

4.3.2.1 Mesh-Material Mapping in Griffin

PROTEUS-MOC utilizes a 2D finite element mesh in conjunction with an axial extrusion file. First, the 2D UFMESH file is converted into ascii format, and then extruded to 3D using the Argonne Mesh Tools utility MT_BuildAxialMesh. The resulting 3D mesh is converted to Griffin format (Exodus) using the ProteusAsciiMesh mesh converter of Griffin with the "--mesh-only" option.

This utility also produces the mapping between 3D block locations and block IDs in the 3D exodus mesh.

By referring to the block-material mapping in the assignment file of PROTEUS and using the relation of blocks in the 3D ascii mesh and the 3D exodus mesh, the script establishes the block-material mapping in the 3D exodus mesh, which is finally written in the input block of the SubdomainElementIDs mesh generator in Griffin.

4.3.2.2 Conversion of XS to ISOXML format

The microscopic cross section file in ISOXML format is prepared by calling the ISOXML module with an input ISOTXS file. At the time of developing the python script, there was no function in the ISOXML module to generate the EnergyDeposition variable used to calculate power in the conversion of an ISOTXS file to an ISOXML file. Thus, the script prepares the EnergyDeposition variable for all isotopes from $Q_{fission}$ and $Q_{capture}$ values and cross sections in the ISOTXS file using the same method (HMG4C) as PROTEUS. In this way, power at non-fueled regions like cladding can be calculated by considering gamma energy produced by (n,γ), (n,p), (n,d), (n,t) and (n,α) reactions and using the assumption of local energy deposition. Later in the year, the EnergyDeposition variable was added to the ISOXML module, but the definition of it was different from the definition that PROTEUS uses and there has been a discussion on its exact definition that has not been settled and merged yet.

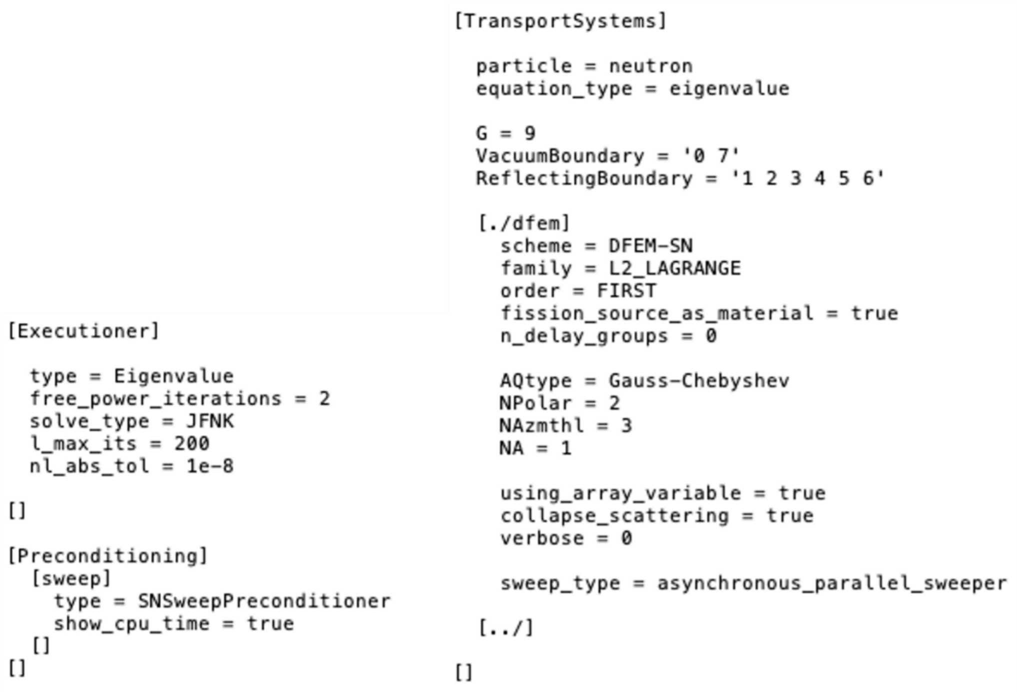
5 Griffin Results and Performance

This chapter describes efforts to determine optimal solver options, mesh and angular discretizations, and accuracy for the nominal condition of the LFR single assembly problem. All calculations used 9-group cross sections due to the current memory requirements in Griffin for 33-group calculations.

5.1 Transport Solver Options

Griffin currently supports three finite-element based discretization schemes for neutronics: (1) the self-adjoint angular flux continuous finite element method (SAAF-CFEM), (2) the least square continuous finite element method (LS-CFEM) and (3) the discontinuous finite element method (DFEM) stabilized using upwinding. Griffin supports either discrete ordinates (SN) or spherical harmonics (PN) angular discretization.

For advanced hot channel factor calculations, the pin geometry is represented heterogeneously. The axial pin power distribution is the most important quantity to calculate neutronically as this distribution is passed to an advanced thermal fluids solver for the temperature and velocity calculation. The DFEM method with SN angular discretization was targeted for performance analysis, as this scheme is well suited to problems with significant material heterogeneity and streaming regions like those present in the annular LFR fuel. A sample input for performing an unaccelerated (direct) DFEM-SN transport solve is shown in Figure 5.1.



```
[Executioner]

  type = Eigenvalue
  free_power_iterations = 2
  solve_type = JFNK
  l_max_its = 200
  nl_abs_tol = 1e-8

[]

[Preconditioning]
  [sweep]
    type = SNSweepPreconditioner
    show_cpu_time = true
  []
[]
```

```
[TransportSystems]

  particle = neutron
  equation_type = eigenvalue

  G = 9
  VacuumBoundary = '0 7'
  ReflectingBoundary = '1 2 3 4 5 6'

  [./dfem]
    scheme = DFEM-SN
    family = L2_LAGRANGE
    order = FIRST
    fission_source_as_material = true
    n_delay_groups = 0

    AQtype = Gauss-Chebyshev
    NPolar = 2
    NAzmthl = 3
    NA = 1

    using_array_variable = true
    collapse_scattering = true
    verbose = 0

    sweep_type = asynchronous_parallel_sweeper

  [../]

[]
```

Figure 5.1. Sample input for unaccelerated (direct) solve using DFEM-SN.

This input sample is largely based on the Griffin tutorial for implementing direct DFEM-SN for solving a 2D version of the C5G7 reactor problem. The inputs are configured to execute an eigenvalue solve with the `SNSweepPreconditioner`. An optional optimization is available in

Griffin making use of array variable objects native to MOOSE – this option was used for all DFEM-SN simulations in this work. This optimization requires setting the `using_array_variable` and `collapse_scattering` values in the `TransportSystems` block to true. These options can be considered standard for an unaccelerated transport solve.

5.2 Diffusion-Based Acceleration

Griffin also supports diffusion-based acceleration – primarily, the nonlinear diffusion acceleration (NDA) method with various closure schemes. Optionally, a diffusion synthetic acceleration-based (DSA) stabilizing scheme can be applied with the NDA accelerator, which operates by lagging closure terms. This can improve the conditioning of the diffusion system for problems that are difficult to converge stably. For the DFEM discretization, both coarse- and fine-mesh diffusion acceleration is possible. Since an additional acceleration scheme, CMFD (Coarse Mesh Finite Difference), was also developed at the time of writing, in total six DFEM solver methods were considered: unaccelerated (direct), three coarse-mesh accelerations (CM-NDA, CM-NDA+DSA,CMFD) and two fine-mesh accelerations (FM-NDA, FM-NDA+DSA). Due to release of the optimized CMFD option late in FY21, CMFD could not be extensively explored for our application. Mesh and angular quadrature sensitivity test results of the direct solver and NDA acceleration are presented first, followed by a brief consideration of CMFD results.

5.2.1 Diffusion Mesh

As discussed previously, the transport mesh is a fine mesh with explicit pin geometry and ducts, generated using the Argonne Mesh Tools package and converted to Exodus format for compatibility with Griffin.

In addition to the fine mesh used in the transport simulation, a suitable coarse mesh was required for the coarse-mesh acceleration options (CM-NDA, CM-NDA+DSA). Several mesh options were considered with varying degrees of geometric fidelity. A section of the final coarse mesh is shown in Figure 5.2. This mesh uses the same axial extrusion as the transport fine mesh and was created with coarse hexagonal cells in the interior of the assembly, without explicitly modeling the fuel pin geometry. In the outer ring, it was necessary to include minimally representative geometry for the fuel pins - a single circular region. Without this, the Argonne Mesh Tools generator would not create cells near the outer assembly duct that were properly nested within the transport mesh. This coarse mesh provided a good balance of stability and computational efficiency for the nominal condition problem. For all fine-mesh methods (FM-NDA, FM-NDA+DSA), the same mesh was used for both diffusion and transport.

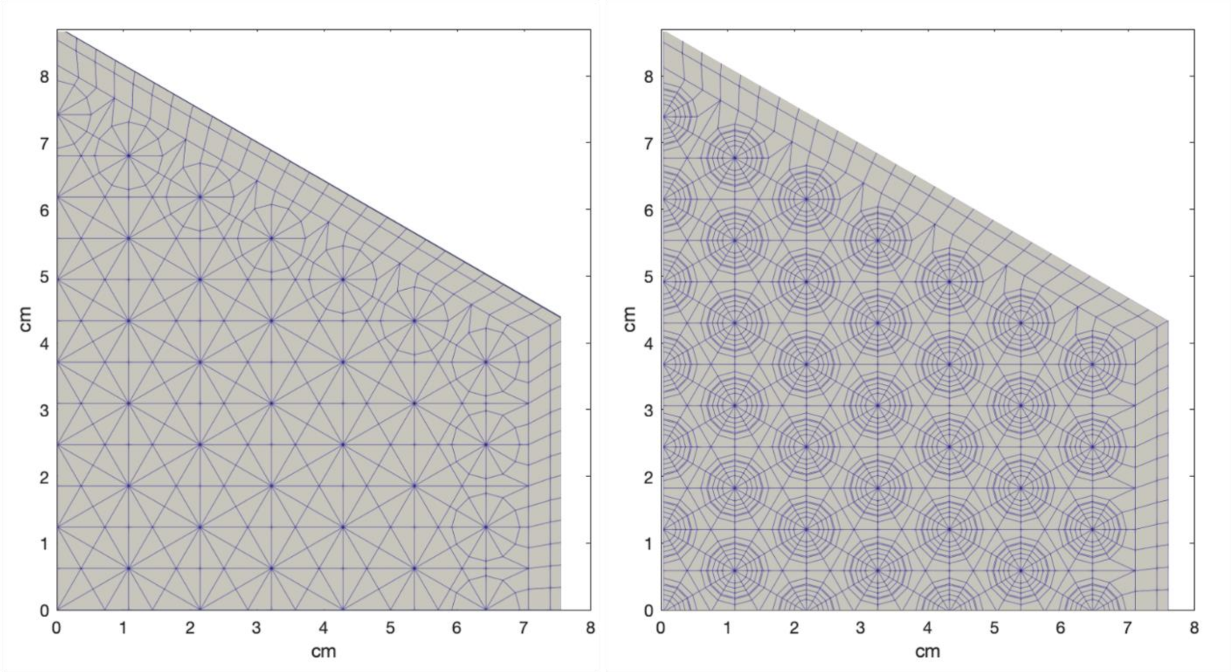


Figure 5.2. Meshes used in diffusion-accelerated DFEM-SN: (left) Coarse diffusion mesh used in CM-NDA and CM-NDA+DSA, and (right) fine mesh used for neutron transport, as well as for diffusion in FM-DSA and FM-NDA.

5.2.2 Input Setup via MOOSE's MultiApp System

Diffusion-based acceleration in Griffin is implemented using the hierarchical, multi-application structure native to the MOOSE framework. This system is designed to enable efficient, parallel solution of coupled PDEs by simultaneously solving individual, coupled physical systems. In a MultiApp problem, the primary solve is conducted by the "main" application, which spawns sub-applications for other physics as specified within the input file. For more complicated multi-physics simulations, this can be done hierarchically, such that each sub-application can spawn lower-level processes for physics relevant only to that process.

NDA can be mapped onto this framework, as the diffusion and transport problems are coupled through restriction of the angular flux variable as well as flux-corrected cross sections in the diffusion problem. An advantage of this implementation is that it allows for re-use of input files from individual diffusion and transport problems, only requiring some changes in the `Executioner` and `TransportSystems` blocks. In the current implementation, the diffusion application is treated as the main app, and it spawns a transport sub-application to perform transport updates. Syntax samples for implementing NDA+DSA in Griffin for the 3D LFR problem are shown in Figure 5.3. These samples are largely based on the Griffin tutorial on implementing DFEM-SN with NDA for a 2D version of the C5G7 reactor.

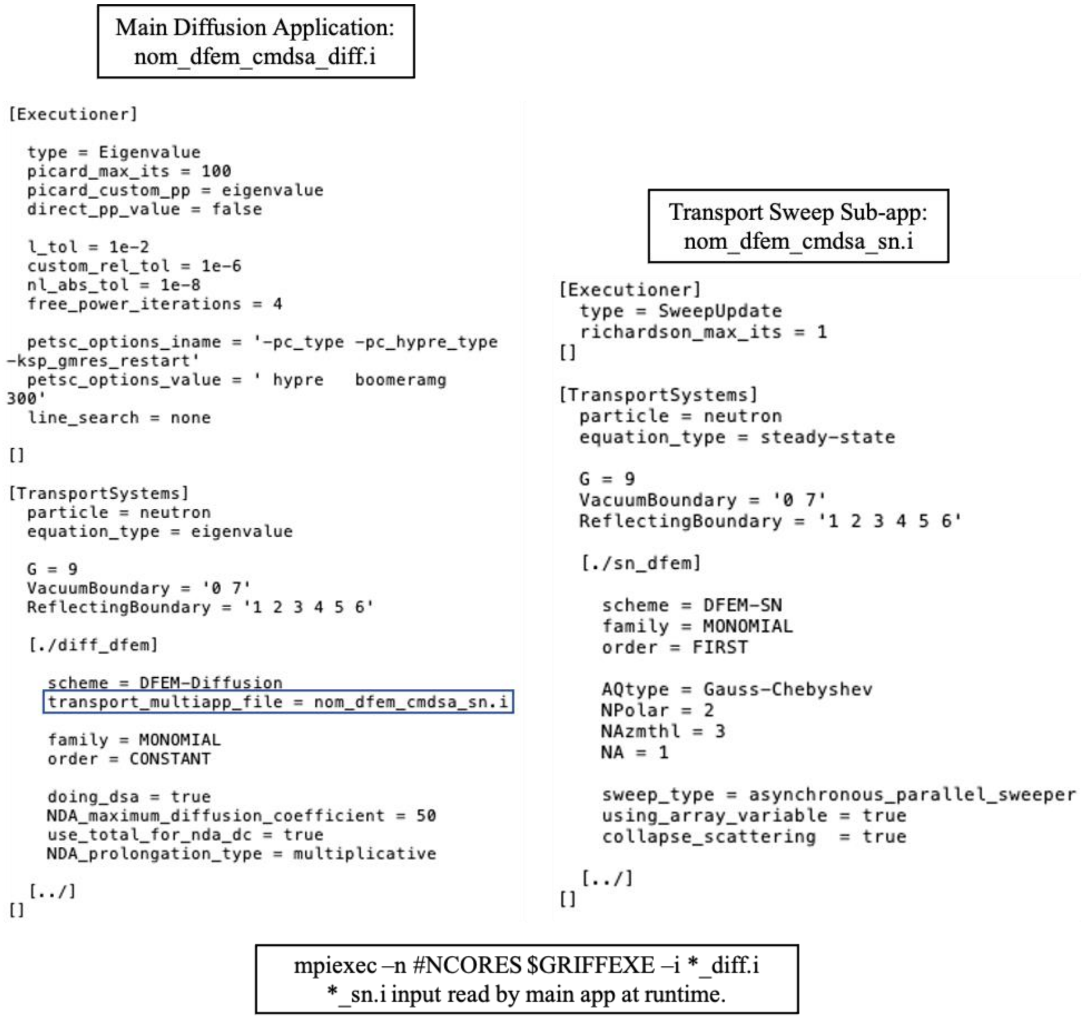
The transport input for diffusion-based acceleration differs in a few ways from the input for a direct solve. The Executioner/type is changed from `Eigenvalue` to `SweepUpdate`, which declares that it is to be used as a transport sweeping sub-process of the diffusion solve. The option `richardson_max_its` can be used to set the number of transport sweeps to perform between Picard iterations – in this case, a single sweep was used. The `SNSweepPreconditioner` is removed entirely, as preconditioning will primarily occur in the diffusion solve. In the `TransportSystems` block, the equation_type is changed from eigenvalue to steady-state, such that the fission source transferred from the diffusion application is treated as a fixed-source. Otherwise, the input is set up identically to the input used to implement unaccelerated DFEM-SN.

In the Executioner block of the diffusion input, an eigenvalue problem is declared. Griffin allows the use of several custom post-processors for use as a convergence criteria – in this input, the change in eigenvalue is used for the convergence check during the Picard iteration. This is a typical choice, and was borrowed from the Griffin C5G7 tutorial. Hypre BoomerAMG preconditioning is used for the diffusion solve.

In the TransportSystems block, an eigenvalue problem is declared with the DFEM-Diffusion discretization scheme. The transport_multiapp_file input option is used to provide a path to the input file for the transport sub-application. The option NDA_maximum_diffusion_coefficient is used to address instability due to near-void regions – as previously noted, the helium-filled regions of the fuel have very small total interaction cross-sections, which can result in extremely large diffusion coefficients. If the calculated diffusion coefficient would exceed the value of NDA_maximum_diffusion_coefficient, the value is instead set to this maximum. The option use_total_for_nda_dc declares whether the total cross section should be used in calculation of the diffusion coefficient. Finally, the NDA_prolongation_type option declares the type of prolongation operator used for updating the angular flux in the transport solve – in this case, multiplicative prolongation of the scalar flux is used.

5.2.3 Input Setup for CMFD

One of the biggest advantages of CMFD is that fine and coarse meshes are read in one input as shown in Figure 5.4, which means the CMFD itself works on a single app, not on Multiapps like the NDA acceleration. This makes only the CMFD solver among different acceleration solvers compatible with the current architecture of the Stochastic Tools that will be briefly explained in Section 8. The basic setup of Executioner and TransportSystems is the same with that of the sub-app input for NDA acceleration except that the equation_type of TransportSystems is “eigenvalue”. On top of that, CMFD-related input cards should be specified in the TransportSystems block. It should be noted that cmfd_solver_type should be “newton” for problems with large number of meshes. The “newton” option is in general faster for large meshes, and “krylovshur” is more suitable for smaller number of meshes and more stable but a little bit slower than “newton”. For our application, “krylovshur” took about 2.5 days to solve the problem, while “newton” took about several tens of minutes using the same options. Detailed results will be provided in a later section. All the results are based on the “newton” solver type.



Main Diffusion Application:
nom_dfem_cmdsa_diff.i

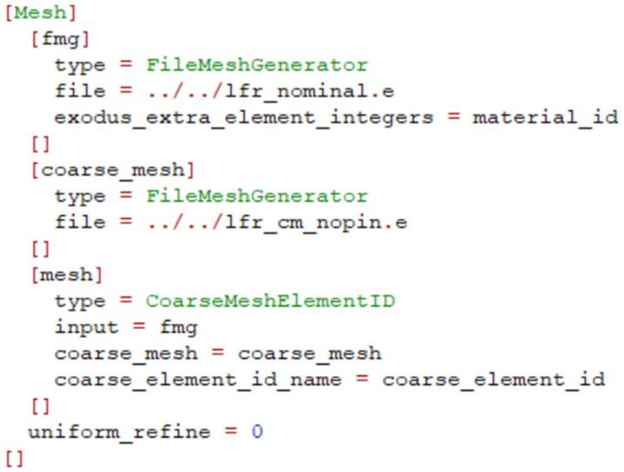
```
[Executioner]

  type = Eigenvalue
  picard_max_its = 100
  picard_custom_pp = eigenvalue
  direct_pp_value = false

  l_tol = 1e-2
  custom_rel_tol = 1e-6
  nl_abs_tol = 1e-8
  free_power_iterations = 4

  petsc_options_iname = '-pc_type -pc_hypre_type
-ksp_gmres_restart'
  petsc_options_value = ' hypre   boomeramg
300'
  line_search = none

[]

[TransportSystems]
  particle = neutron
  equation_type = eigenvalue

  G = 9
  VacuumBoundary = '0 7'
  ReflectingBoundary = '1 2 3 4 5 6'

  [./diff_dfem]

    scheme = DFEM-Diffusion
    transport_multiapp_file = nom_dfem_cmdsa_sn.i

    family = MONOMIAL
    order = CONSTANT

    doing_dsa = true
    NDA_maximum_diffusion_coefficient = 50
    use_total_for_nda_dc = true
    NDA_prolongation_type = multiplicative

  [../]
[]
```

Transport Sweep Sub-app:
nom_dfem_cmdsa_sn.i

```
[Executioner]
  type = SweepUpdate
  richardson_max_its = 1
[]

[TransportSystems]
  particle = neutron
  equation_type = steady-state

  G = 9
  VacuumBoundary = '0 7'
  ReflectingBoundary = '1 2 3 4 5 6'

  [./sn_dfem]

    scheme = DFEM-SN
    family = MONOMIAL
    order = FIRST

    AQtype = Gauss-Chebyshev
    NPolar = 2
    NAzmthl = 3
    NA = 1

    sweep_type = asynchronous_parallel_sweeper
    using_array_variable = true
    collapse_scattering  = true

  [../]
[]
```

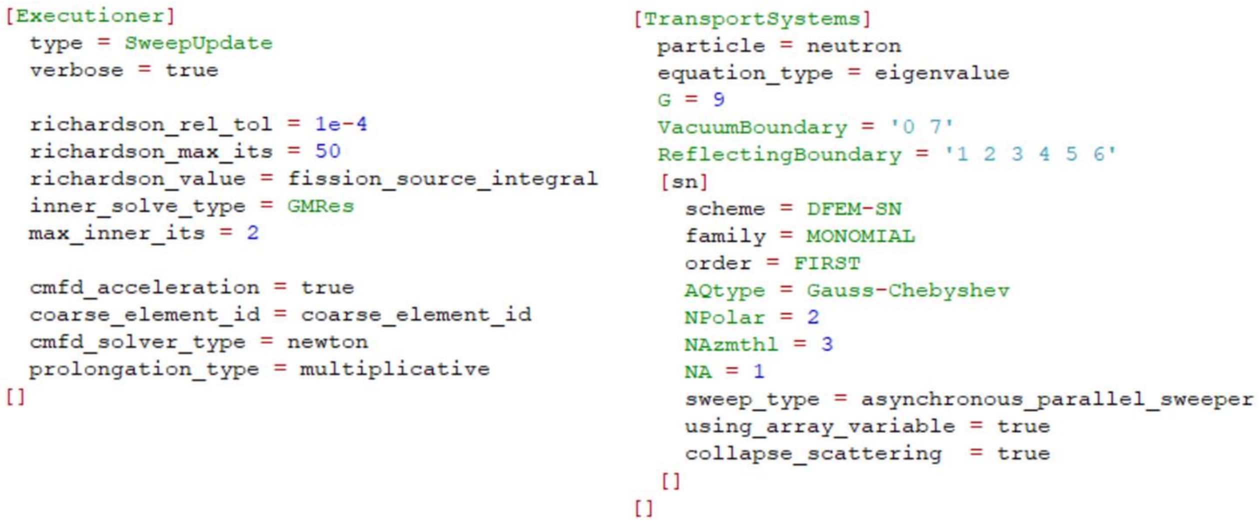
mpiexec –n #NCORES $GRIFFEXE –i *_diff.i
*_sn.i input read by main app at runtime.

Figure 5.3. Sample hierarchical multi-application input syntax used in Griffin for implementing coarse-mesh DSA with DFEM-SN transport updates.



```
[Mesh]
  [fmg]
    type = FileMeshGenerator
    file = ../../lfr_nominal.e
    exodus_extra_element_integers = material_id
  []
  [coarse_mesh]
    type = FileMeshGenerator
    file = ../../lfr_cm_nopin.e
  []
  [mesh]
    type = CoarseMeshElementID
    input = fmg
    coarse_mesh = coarse_mesh
    coarse_element_id_name = coarse_element_id
  []
  uniform_refine = 0
[]
```

Figure 5.4. Sample mesh block for CMFD.



```
[Executioner]
  type = SweepUpdate
  verbose = true

  richardson_rel_tol = 1e-4
  richardson_max_its = 50
  richardson_value = fission_source_integral
  inner_solve_type = GMRes
  max_inner_its = 2

  cmfd_acceleration = true
  coarse_element_id = coarse_element_id
  cmfd_solver_type = newton
  prolongation_type = multiplicative
[]
```

```
[TransportSystems]
  particle = neutron
  equation_type = eigenvalue
  G = 9
  VacuumBoundary = '0 7'
  ReflectingBoundary = '1 2 3 4 5 6'
  [sn]
    scheme = DFEM-SN
    family = MONOMIAL
    order = FIRST
    AQtype = Gauss-Chebyshev
    NPolar = 2
    NAzmthl = 3
    NA = 1
    sweep_type = asynchronous_parallel_sweeper
    using_array_variable = true
    collapse_scattering  = true
  []
[]
```

Figure 5.5. Sample Executioner and TransportSystems blocks for CMFD.

5.3 Computational Hardware

For all cases considered in this performance study, calculations were performed on the Broadwell partition on Argonne's Laboratory Computing Resource Center (LCRC) Bebop machine which avails up to 664 Intel-Xeon e52695v4 nodes (36 cores per node, 128 GB DDR4 memory per node). All calculations used 32 nodes, utilizing only 18 of the 36 available cores per node for a total of 576 cores and 4 TB memory. We note that half the cores were idled to increase the memory available to each compute core. This was found to be necessary, as further parallelizing the problem to use more nodes did not reduce the memory required per core.

5.4 Performance Metrics

We considered two metrics for assessing the performance of each set of solver options: (1) total wall-clock runtime and (2) average memory-per-processor. Runtime is an important optimization for hot channel factor problems, as collecting adequate data may require fully solving many perturbed cases. The 3D nominal condition problem is relatively memory intensive to solve, as observed both from prior years using PROTEUS and in early efforts running the simulation in Griffin. Because memory can be limiting for even the single physics neutronics problem, this was chosen as a metric for comparing solver options.

5.5 Space-Angle Convergence Studies

DFEM-SN methods were used for the following studies: unaccelerated (direct), FM-NDA, FM-NDA+DSA, CM-NDA, and CM-NDA+DSA. All simulations utilized the 9-group cross sections generated using the 1-D radial model with one axial cross section zone with axial leakage considered as discussed in 4.2.2.

5.5.1 Angular Refinement Studies

Recalling back to previous efforts using PROTEUS-MOC, the 3D LFR assembly case exhibited a strong dependence on the degree of angular cubature due to the presence of many streaming regions in the annular fuel geometry. The converged angular cubature for the PROTEUS-MOC method was Legendre-Tchebyshev L7T17 (288 angles). The Griffin DFEM-SN simulations employ Gauss-Chebyshev discrete ordinates cubature (Griffin also supports other angular discretizations that may be explored in future study). The Gauss-Chebyshev cubature is defined by the number of polar (P) and azimuthal (A) directions used in each octant, such that the total number of discrete ordinates is N_O = 8P*A. Six angular refinements were considered and are summarized in Table 5.1.

Input files for each of these cases were created for each solver method, using the mesh shown to be fully converged with PROTEUS-MOC in previous work. The updated `HexagonalGridVariableIntegral` post-processor described in Section 4.3.1.2 was used to integrate the power in the center fuel pin across 20 uniform axial slices to directly compare Griffin axial powers with MCNP tallied axial powers.

Table 5.1. Angular discretizations considered for sensitivity study.

# Polar Angles (P)	1	2	2	3	3	4
# Azimuthal Angles (A)	3	3	6	6	12	12
# Ordinates (N_O)	24	48	96	144	288	384

Figure 5.6 displays eigenvalue (left) and axial power RMS error (right) convergence with increasing angular refinement from DFEM-NDA+DSA-CM. The right figure specifically shows the RMS error between the axial power and the MCNP solution. The various Griffin solver methods produced results with consistent eigenvalues and axial power distributions as the angular cubature was refined, so these results are representative for all methods included in the study.

As anticipated from experience with PROTEUS, sensitivity to the angular cubature for this problem was also observed with Griffin. Between the P1A3 (24 angles) and P4A12 (384 angles) cubature sets, the resulting eigenvalue changed by 120 pcm, settling on a value k_{eff}=1.17237. This eigenvalue agrees well (+68 pcm) with continuous energy MCNP ($k_{eff,MCNP}$ = 1.17169).

The axial power was less sensitive to the angular discretization than the eigenvalue. The RMS error between the Griffin and MCNP axial powers rapidly reaches a steady magnitude (around 0.49 in RMS % error, 1.25% max, -0.51% min), with a relatively minor difference between the P2A3 (48 angles) cubature set and all higher refinements. Because of the energy discretization differences between continuous energy MCNP and 9-group Griffin, some error in the 9-group deterministic Griffin solution is expected even when fully converged in space and angle. This bias is observed in

Figure 5.6 (right), where the deterministic error increases with angular refinement until the 9-group Griffin solution has converged.

Further examining the actual power shapes rather than just looking at the integrated error, Figure 5.7 contains plots of the axial power distribution for the highest angular refinement (P4A12) against the MCNP power (left), as well as plots of the error distribution for each cubature (right). The relative error reaches around 1.2% in the top/bottom core regions, and at around -0.6% near the core center. This is consistent with the results obtained with PROTEUS as shown by the green color of the right top plot in Figure 4.3. As discussed in Section 4.2.2, this error behavior is dictated by the accuracy of multigroup cross section set (influenced by group structure among other uncertainties), and it should be confirmed that how large of an impact these deviations have on the thermal fluids calculation. Additional discussion is located in Section 4.2.3.2.

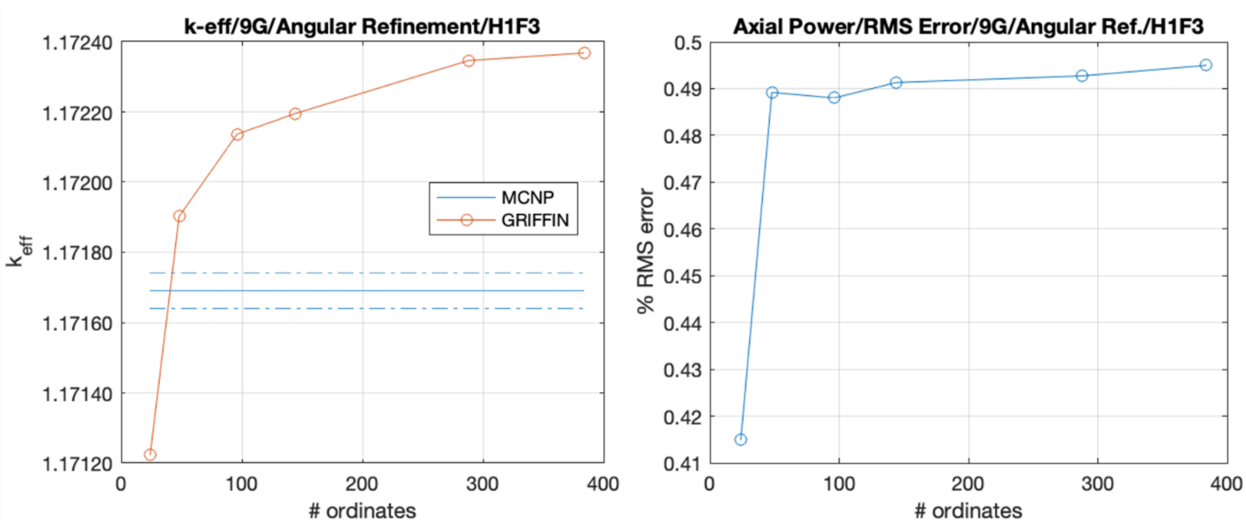


Figure 5.6. Convergence plots from angular refinement study. Left: k-eigenvalue, converging to a value roughly 67 pcm different from the MCNP reference solution (1.17169 with 1σ error bars). Right: RMS error between Griffin axial power and MCNP reference solution.

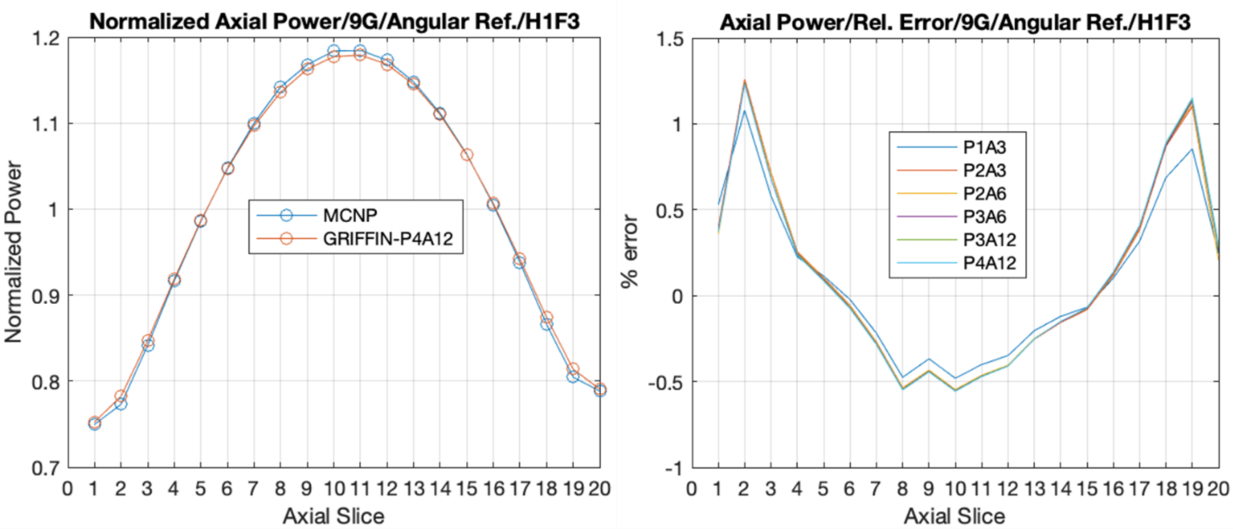


Figure 5.7. Axial power distribution compared with MCNP. Left: normalized axial power of highest angular refinement (P4A12). Right: relative error distributions between Griffin results and the MCNP reference solution.

In conclusion, angular convergence with Griffin is similar to that shown with PROTEUS-MOC: the 288 ordinate cubature (P3A12) is required if both eigenvalue and axial power are required to be converged, but lower cubatures (P2A3, 48 angles or P2A6, 96 angles) are acceptable in terms of power convergence which is key for these hot channel factor simulations.

5.5.2 Angular Refinement Performance

Figure 5.8 displays the wall clock run time and average memory per process for each solver type with increasing angular refinement. There was very little difference in either metric between the two coarse mesh methods (CM-NDA, CM-NDA+DSA), as well as between the two fine mesh methods (FM-NDA, FM-NDA+DSA).

At this time, very little data is available for the unaccelerated DFEM-SN solver method. This is due to memory limitations - the cause is currently unclear, but the memory requirement for unaccelerated DFEM-SN increases unpredictably as the problem size increases. This made it impossible to use unaccelerated DFEM-SN to solve higher angular refinements when using the 32 node/576 core configuration on Broadwell. Regardless, it is clear that the accelerated methods outperform the direct solve for the nominal condition problem, so further attempts at using unaccelerated DFEM-SN were not pursued.

For the P2A3 cubature, on the nominal converged PROTEUS-MOC mesh, the unaccelerated solution ran for 2200 seconds, whereas the fine-mesh acceleration methods required roughly 1360 seconds and the coarse-mesh acceleration methods required roughly 670 seconds. This is a factor of roughly 1.63 improved with FM-NDA/FM-NDA+DSA, and 3.28 using CM-NDA/CM-NDA+DSA. The coarse-mesh methods were significantly faster than all other approaches, which is valuable for efficiently simulating perturbed cases.

The difference in memory usage between fine- and coarse-mesh methods was not very large, but there was an interesting trend. At lower angular refinements, the coarse-mesh methods used less memory than fine-mesh methods, but beyond a certain problem size (in angle) the coarse-mesh methods became more memory intensive. However, the sensitivity study showed that 48-96 angles (P2A3, P2A6) were sufficient for convergence in the axial power distribution. For these cubatures, the coarse mesh acceleration options do not consume excessive memory resources.

There did not appear to be stability issues with the nominal condition problem, indicating an additional stabilizer was not needed. However, between the coarse mesh accelerated methods, the DSA stabilizing option did not have a significant impact on memory consumption. There was a small amount of slowdown, likely due to an increase in overhead, but the overall convergence rate did not appear to be significantly slowed. This result could be invaluable if difficult-to-converge perturbed cases are encountered in the future, since DSA stabilization can be used without significant performance cost.

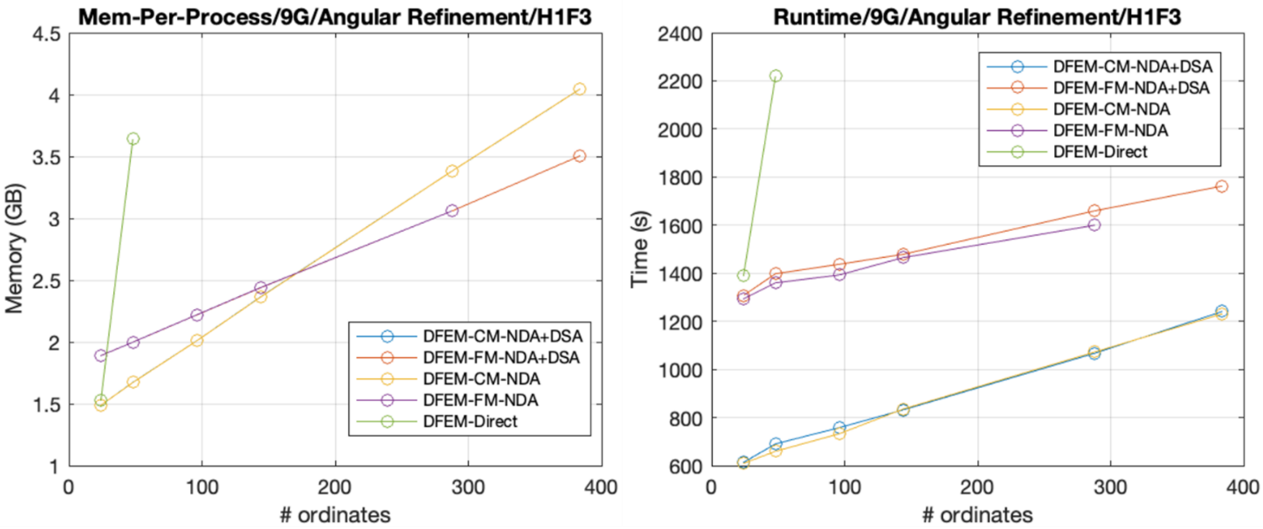


Figure 5.8. Performance metrics for each solver type with angular refinement. Left: average memory per process, Right: total runtime.

5.5.3 *Spatial Refinement Studies*

A spatial convergence study was conducted to identify the optimal radial refinement. Typically, a study of this type should consider the effects of both radial and axial refinements. However, with the current workflow for mesh generation in PROTEUS and material assignment in Griffin, it was difficult to implement axial refinement studies which introduce additional subplanes targeted at the bottom/top of the core. Further work should be done to improve the quality of the axial mesh, as this has been identified as a promising approach for reducing overall error in the solution from Griffin.

The nominal converged mesh from PROTEUS-MOC simulations was used as a starting basis. A radial refinement study was conducted by adding regions to both the fueled and helium-filled regions of the fuel elements, of which the nominal converged mesh had 1 radial zone in the helium regions and 3 radial zones in the fuel (H=1, F=3, or H1F3). Figure 5.7 shows an example of this additive refinement for a single fuel pin, from H1F3 to H2F4. Six meshes were created with increasing refinement in these regions. An additional, coarser mesh was included (H1F1) to observe the improvement of the eigenvalue and power up to the nominally converged mesh case. The discretizations considered are summarized in Figure 5.2.

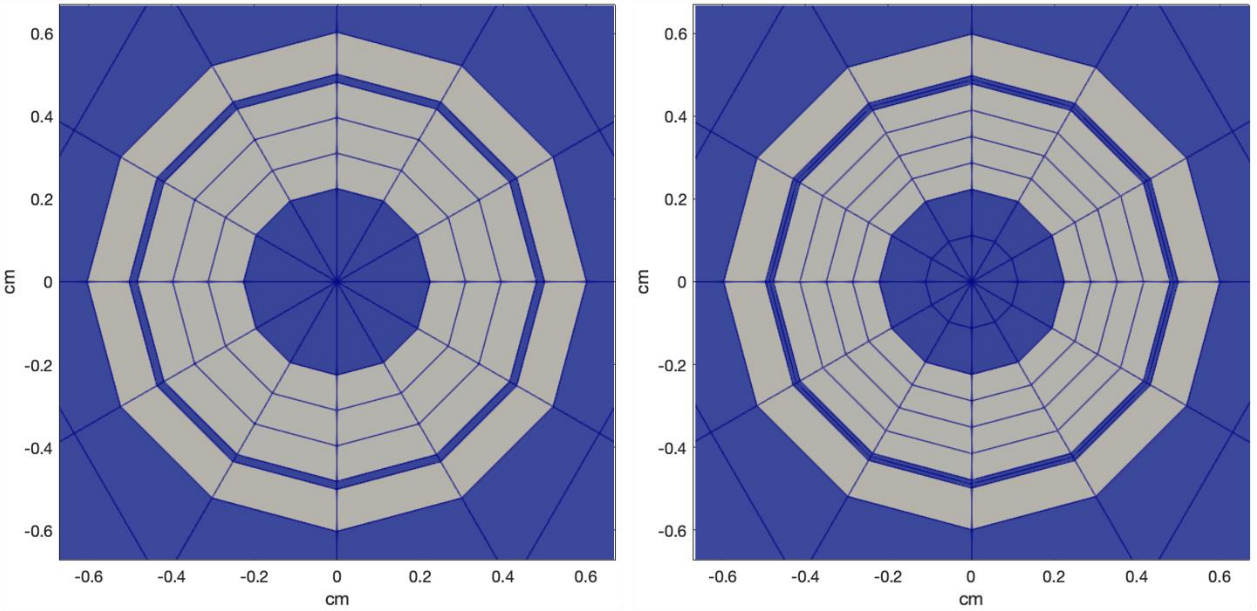


Figure 5.7. Comparison of single-pin meshes used in two different radial refinements. Left: single-pin with 1 radial zone in the void region and 3 radial zones in the fuel (H1F3). Right: single-pin with 2 radial zones in the void region and 4 radial zones in the fuel (H2F4).

Table 5.2. Mesh discretizations considered for sensitivity study.

# Void Radial Regions (H)	1	1	2	4	4	4	6
# Fuel Radial Regions (F)	1	3	4	6	8	12	16
# Mesh Nodes (3D)	3.55E5	5.05E5	7.29E5	1.33E6	1.48E6	1.77E6	2.37E6

The nominal condition problem was very insensitive to these changes in the radial discretization. Figure 5.9 shows convergence histories for both the eigenvalue and the RMS error in the axial power. Across all discretizations, the eigenvalue settles to approximately 1.17189 (+20 pcm from continuous energy MCNP). There is negligible improvement increasing the mesh refinement from 3.55E+05 nodes to 2.37E+06 nodes, indicating that the lowest mesh refinement is enough for the solution to be fully converged in space. This observation is reflected in the axial power distribution. Figure 5.10 shows the axial distribution from the highest mesh quality as well as the error distribution for the highest spatial refinement. The distributions from other mesh refinements were excluded, as they are indistinguishable from each other in this plot. It is evident that the nominal condition problem is insensitive to radial refinement beyond the lowest case considered (H1 F1).

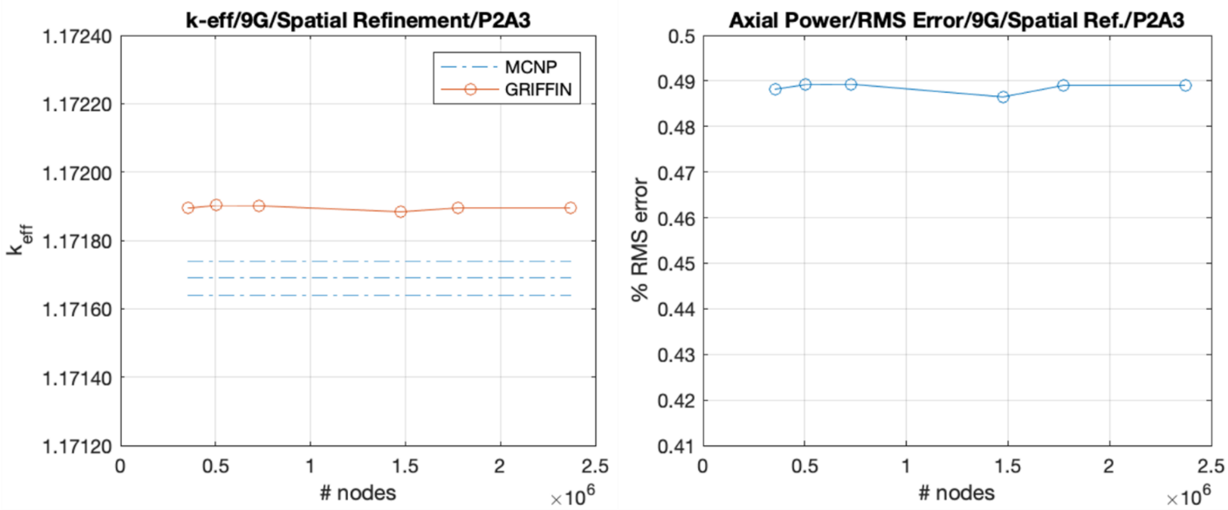


Figure 5.9. Convergence plots from spatial refinement study. Left: k-eigenvalue, converging to a value roughly 18 pcm different from the MCNP reference solution. Right: integrated error between Griffin axial power and MCNP reference solution.

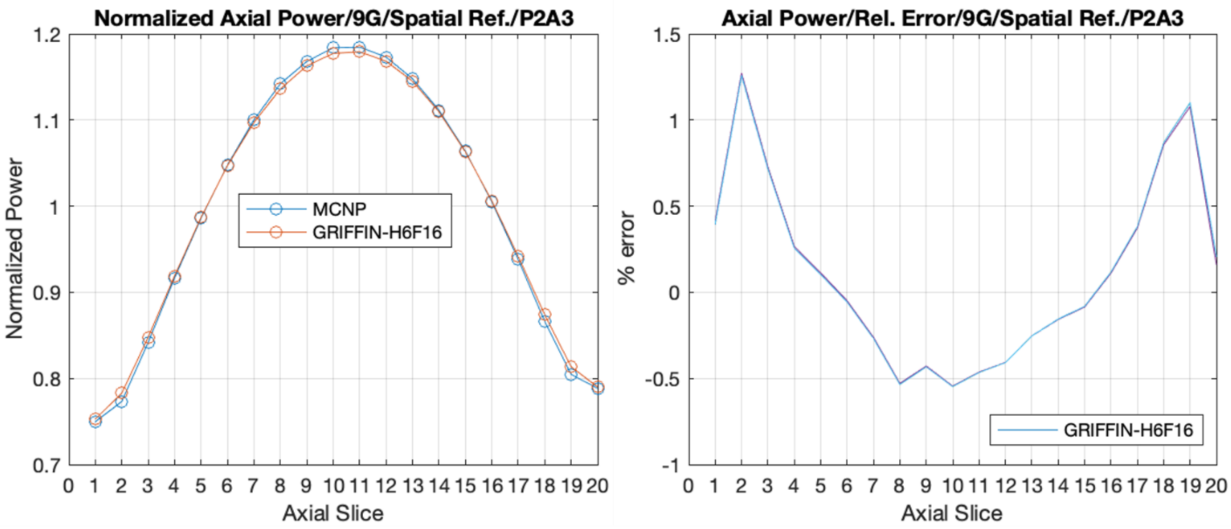


Figure 5.10. Axial power distribution compared with MCNP. Left: normalized axial power of highest spatial refinement (H6F16). Right: relative error distributions between Griffin-H6F16 result and the MCNP reference solution.

5.5.4 Spatial Refinement Performance

Figure 5.9 displays the wall clock run time and average memory per process for each solver type with increasing radial refinement. There was very little difference in either metric between the two coarse mesh methods (CM-NDA, CM-NDA+DSA), as well as between the two fine mesh methods (FM-NDA, FM-NDA+DSA).

As observed in the angular performance study, the unaccelerated DFEM-SN method was limited by memory scaling for larger problem sizes, and it was not possible to run higher problem sizes with the 32 node/576 core configuration on Broadwell.

It was noted in the angular refinement performance study that runtimes were consistently improved by a factor of roughly 1.63 with FM-NDA/FM-NDA+DSA, and 3.28 using CM-NDA/CM-NDA+DSA. This is again observed for the nominal converged mesh with P2A3 cubature, but the trend at higher refinements is much more favorable for the coarse-mesh accelerated methods. For the (H=2, F=4) mesh refinement, the runtime improvement from fine-mesh acceleration was a factor of 1.49, while the improvement with coarse-mesh acceleration was a factor of 3.75. Though unaccelerated data is unavailable beyond this problem size, the runtime gap between fine- and coarse-mesh acceleration grows for larger problem sizes, such that the coarse-mesh methods exhibit more favorable scaling. It is hypothesized that this is due to differences in scaling of the diffusion problem size between the fine- and coarse-mesh methods. Because the diffusion solve uses the same mesh as transport in fine-mesh accelerated methods, the diffusion problem increases in size as the mesh is radially refined. This does not occur for the coarse-mesh acceleration methods (which all use the same coarse mesh shown in Figure 5.2), or even for the fine-mesh methods in the angular refinement case (where the gap appeared to stay consistent as the problem size increased, instead of growing).

The memory usage showed a relatively straightforward trend – the coarse-mesh accelerated methods consistently required less memory than all other methods (unaccelerated, where data is available, and the fine-mesh methods). The gap between coarse- and fine-mesh methods grows as the problem size increases, indicating that the coarse mesh methods should be preferred for the nominal condition problem.

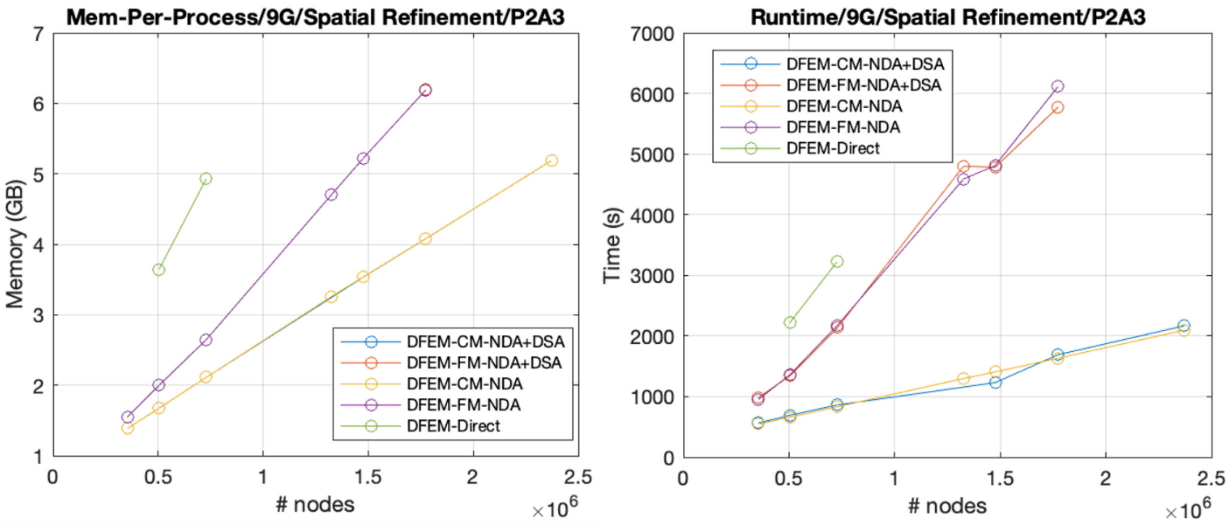


Figure 5.11. Performance metrics for each solver type with spatial refinement. Left: total runtime. Right: average memory per process

5.6 Griffin CMFD Results and Comparison with CM NDA and PROTEUS

Due to a shortage of time to test the CMFD solver at the time of writing, only the coarse angular quadrature set of P2A3 was used for the test and compared the results with other solver options in

Griffin and PROTEUS. For mesh, H1F3 was used, and 9-group cross section of the 1D "1R/Ax. Lkg" case in Table 4.2 was used.

Table 5.3 compares computational performances of CMFD and NDA accelerations of DFEM-SN in Griffin and CMFD acceleration of 2D/3D MOC in PROTEUS. *First, eigenvalue results of Griffin were all identical to that of PROTEUS, and the axial power distribution of Griffin matched very well with that of PROTEUS as shown in Figure 5.12*. For DFEM-SN CMFD results, four different number of MPI processors of 18, 36, 72 and 144 were used. Figure 5.13 shows the computational time taken by CMFD and transport calculations together with the total time. While the transport solve showed very good scalability, slight degradation of parallel efficiency for the CMFD solve led to slight degradation of the parallel efficiency for the total computational time. 13 and 9 CMFD iterations were performed for Griffin and PROTEUS calculations, respectively, and 27 NDA iterations were performed to reach a similar convergence. As a result, the CMFD solver was about 4 times faster than the CM-NDA solver, and about 50% slower than PROTEUS MOC with CMFD acceleration. As for memory, the CMFD option consumed about 75% memory of the CM-NDA option, and about 400% memory of PROTEUS. This large memory consumption should be improved since calculations with a larger number of angles and energy groups couldn't be easily queued up on the Broadwell partition on Bebop at the Argonne LCRC, each node of which has 128 GB DDR4 memory. A large number of nodes are required due to memory limit per node and this requirement should be relieved for Griffin to be favored by users.

Table 5.3. Comparison of Computational Performance of Various Solvers

Transport	Acceleration	# of Proc.	Mem. (GB) /Proc.	Total Mem. (GB)	Wall Time (min)	Nonlinear Iteration
DFEM-SN Griffin	CMFD	18	5.03	90.6	51.0	13
		36	3.35	120.5	26.8	13
		72	2.41	173.2	14.9	13
		144	1.92	276.4	9.9	13
DFEM-SN Griffin	CM-NDA	36	4.43	159.4	133.3	27
		576	1.67	963.8	11.5	27
2D/3D MOC PROTEUS	CMFD	36	0.75	27.1	19.1	9

* P2A3 for Griffin and L3T5 for PROTEUS
* 9-group cross section of the 1D "1R/Ax. Lkg" case in Table 4.3
* K-eff = 1.17190 for all cases: $\Delta\rho$=+15 pcm compared to MCNP6.2

As the CMFD acceleration greatly improved the NDA solver, 33-group calculations were attempted using CMFD acceleration. The same angular quadrature of P2A3 was used. Unfortunately, the CMFD matrix couldn't be solved using the "newton" type solver for fast computation, but could only be solved using the "krylovshur" type solver. However, as noted Section 5.2.3, the "krylovshur" solver option was about 300 times slower than the "newton" type solver for 33-group calculation, resulting in computational time of more than 2 days. The memory consumption was 7.76 GB per processor and total of 279.2 GB using 36 processors, which are about 2.3 times larger than those of 9-group calculation. When using P4A9, the memory consumption was 39.10 GB per processor and total of 1.41 TB using 36 processors. These results indicate the need for stabilizing the CMFD solver and reducing large memory consumption for 33-group calculation in Griffin.

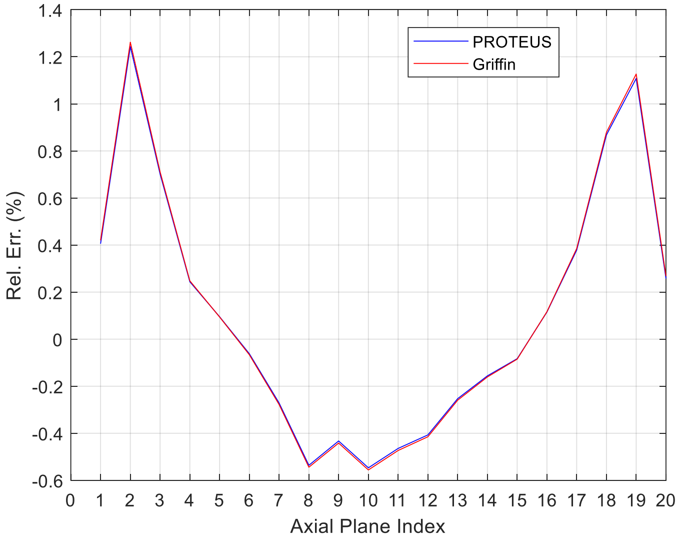


Figure 5.12. Axial power distribution errors of Griffin and PROTEUS results of Table 5.3 compared to MCNP6.2.

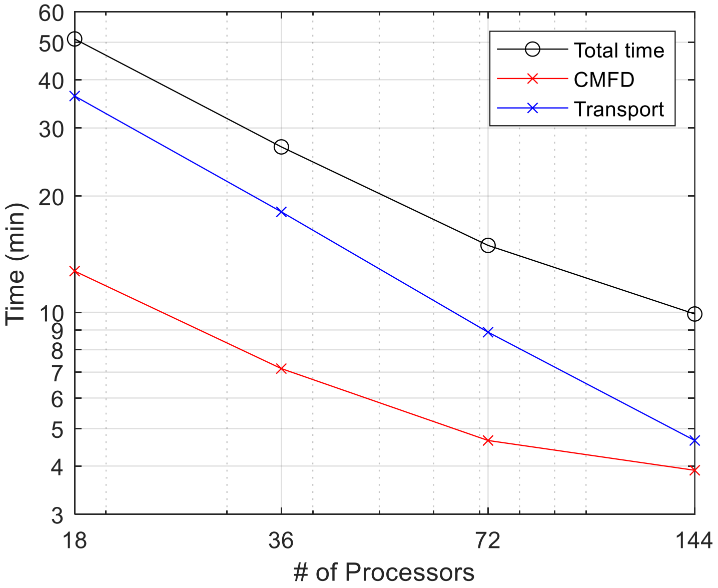


Figure 5.13. Computational time comparison according to the number of processors for DFEM-SN with the CMFD acceleration.

5.7 Griffin Performance Conclusion

The sensitivity study shows that there is little sensitivity to the radial refinement of the mesh. Converged results for the nominal condition problem can be obtained by using either the coarsest mesh (H1F1), or the mesh previously developed for work using PROTEUS-MOC (H1F3). There was significantly stronger sensitivity to the angular cubature, as was observed in previous studies using PROTEUS-MOC. For hot channel factor calculations, convergence in the power distribution is more important than the eigenvalue, and the P2A3 cubature set appears to be sufficient for convergence in power. However, to ensure eigenvalue convergence, P2A6 should be used.

For the NDA, the coarse mesh NDA outperforms the fine mesh NDA. Memory and computational time of the CM NDA were about 25% ~ 50% of those of the FM NDA. DSA can be always applied for stabilization of the NDA since the DSA did not impose noticeable burden on the performance. The CMFD acceleration which was developed at the time of writing turned out to outperform the CM NDA; memory and computational time of the CMFD acceleration were about 75% and 20% of those of the CM NDA, respectively. All these Griffin results were identical to PROTEUS results under the same calculation condition.

The results of the performance study indicate that Griffin can be used for high fidelity neutronics calculation in a practical range of computational time and memory, and thus, for the HCF evaluation work. The performance of the newly developed CMFD acceleration option of Griffin was greatly improved compared to the NDA solver. The computational time and memory of the Griffin CMFD solver and NDA solver are 1.5 and 4 times higher, and 8 and 6 times higher than those of the PROTEUS CMFD solver, respectively, with a comparable accuracy. Using the Griffin CMFD solver, with 2 polar angles and 3 azimuthal angles in an octant that gives practically converged value of power distribution, the 3-D heterogeneous assembly problem that features 48 axial meshes and ~ 1E+4 radial meshes can be solved in about 30 minutes using only 36 processors using 9-group cross section. To perform 33-group calculations with the Griffin CMFD solver, some stability improvements are needed in the CMFD solver, and improvements are also needed in the memory consumption.

5.8 Assessment Summary

Griffin has been demonstrated to run the nominal condition heterogeneous LFR assembly calculation to good agreement (~68 pcm, ~0.5% RMS power) with a reference continuous energy MCNP calculation. Griffin eigenvalue and power distribution were identical to that of PROTEUS-MOC under the same calculation conditions, verifying code-code consistency with the same multigroup cross sections. The following Griffin parameters are currently suggested for this work:

- DFEM-SN solver with the CMFD acceleration to reduce runtime and memory
    - Calculation time and memory consumption were greatly improved compared to the CM NDA: 25% saving in memory and 80% saving in computational time.
    - Compared to PROTEUS-MOC with the CMFD acceleration, the Griffin CMFD takes 4 times more memory and 1.5 times more computational time. However, the performance is in a practical range in an HPC cluster and significant speedups were observed from start to end of FY21 based on Griffin acceleration work. We note that Griffin is not a stand-alone solver and sits on the MOOSE multi-physics framework, so matching the performance of a stand-alone solver is not realistic.
    - We will consult with Griffin developers to determine whether additional input options may save on memory or computational time.
- Angular cubature
    - P2A3 is a practical choice to have a converged power distribution. If eigenvalue should be converged, P2A6 is required at minimum.
- Mesh discretization
    - A relatively coarse transport mesh for DFEM-SN is allowable as the mesh sensitivity study showed little sensitivity with respect to mesh.
- Energy groups
    - 33-group calculations were attempted but could not run due to large memory consumption and instability of the “newton” type solver which prevented the CMFD solve from converging. Therefore, 9-group calculations were performed and studied. We plan to work with Griffin developers to find ways to enable 33-group calculations.
    - MC^2-3 was updated to consider the axial leakage effect on 9-group cross sections for accuracy.
    - Initial multiphysics coupling results (see Section 4.2.3.2) showed weak dependence of HCF on cross section generation, suggesting confidence in using 9-group cross sections for this work, but further studies are needed.

Going forward, MOOSE-based meshing tools being developed in FY21 will be leveraged for easier input preparation of heterogeneous problems. Coordination efforts have already begun and a sample LFR assembly mesh has already been created.

The Griffin source code was updated based on needs of this project:

- Development of MicroNeutronicsMaterial by the Griffin development team

    - No excess memory use by multiple mixing tables with the microscopic cross section approach for keeping accuracy in composition-wise fission spectrum important for fast spectrum reactor
    - Decoupling of material ids assigned in a mesh to those listed in the ISOXML file.
- Update of HexagonalGridVariableIntegral VectorPostProcessor
    - Allow axial mesh-wise power tally within the same block
    - Much more convenient block restriction for tally by user-specified extra element integer like material id

Additional feedback is provided for Griffin developers:

- *Power normalization*: The current version of Griffin normalizes power with respect to kappa-fission, which only considers fission-based heating when ISOXML is generated by the conversion from ISOTXS. This work, and others, requires consideration of non-fission based heating including energy deposition in structural materials resulting from capture reactions. It is unclear how the user should normalize the power to include these non-fission types of heating.
- *Memory issues with larger energy group structures*: Large memory consumption in 33-group calculation was a major hurdle to test Griffin even in a large HPC cluster. Memory burden should be reduced to enable testing at larger group structures. This is the main reason for 9-group calculation in this work.
- *CMFD instability*: Instability of the “newton” type solver prevented the 33-group CMFD solve from converging. The use of “krylovshur’ type solver for stability was 300 times slower than the “newton” type solver. This indicates the need for further improvements in the CMFD solver.

Overall, our assessment is that Griffin is ready for advanced HCF simulations of ducted-assembly fast reactor designs. Addressing the issues in the previous paragraph would further enable robustness and user convenience.

6 NekRS Assessment

This report aims at assessing the capability of the NekRS thermal hydraulic code for HCF calculation. In previous work, Nek5000, which has been developed and extensively applied for many applications for decades, was employed as the thermal-hydraulic module for HCF calculations. NekRS is C++ based, GPU-compatible, MOOSE-wrapped, and offers significant advantages for high performance multiphysics coupling going forward. Given that NekRS is a prototype code, issues in the assessment were anticipated. One issue is that the code is evolving quickly so capabilities sometimes change from one version to another without being backward compatible. NekRS is written in C++ while Nek5000 is written in C and FORTRAN. The setup of each model for exercising a given capability can be completely different. Therefore, the assessment of capabilities in NekRS that are essential for HCF calculations is a significant undertaking. In order to achieve a comprehensive assessment, the code was applied for several applications designed for testing certain capabilities of the code. To make these cases more realistic, all the geometric and flow parameters for these applications were chosen as close to that of the WEC LFR as possible. The results from NekRS were compared with results from Nek5000 to verify these capabilities by checking if identical results for both codes are observed. The complexity of the applications ranges from a simple tube to 7 pin rod bundle. The detailed information and purpose of each case are described in the following sections.

6.1 Introduction on NekRS

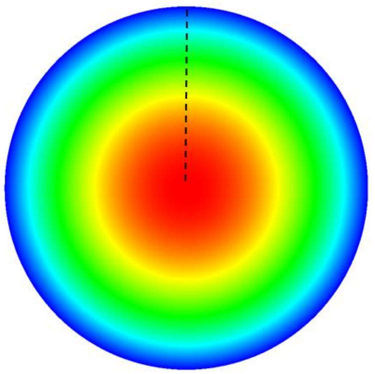
NekRS, a GPU-oriented thermal-fluids simulation code based on the spectral element method (SEM), is developed at Argonne National Laboratory (ANL), University of Illinois-Urbana Champaign (UIUC), and Pennyslvania State University (PSU). With current exascale computing programs in the U.S. and elsewhere developing GPU-based HPC platforms it is imperative to exploit the performance potential of these powerful node architectures. NekRS aims to leverage the present trend in GPU-based HPC systems to perform CFD on GPU-accelerated systems. By using the OCCA [19,20] library's unified API, NekRS can run on CPUs and on GPU-accelerated CPUs that support CUDA, HIP, or OpenCL. For performance portability, the code is based on the open concurrent compute abstraction and leverages scalable developments in the SEM code Nek5000 and in libParanumal [21,22], which is a library of high-performance kernels for high-order discretization and PDE-based mini-apps. Critical performance results on several platforms indicates the strong scaling of NekRS including scaling to 27,648 V100s on OLCF Summit, for calculations of up to 60B grid points [23].

6.2 LES Simulation on a Tube

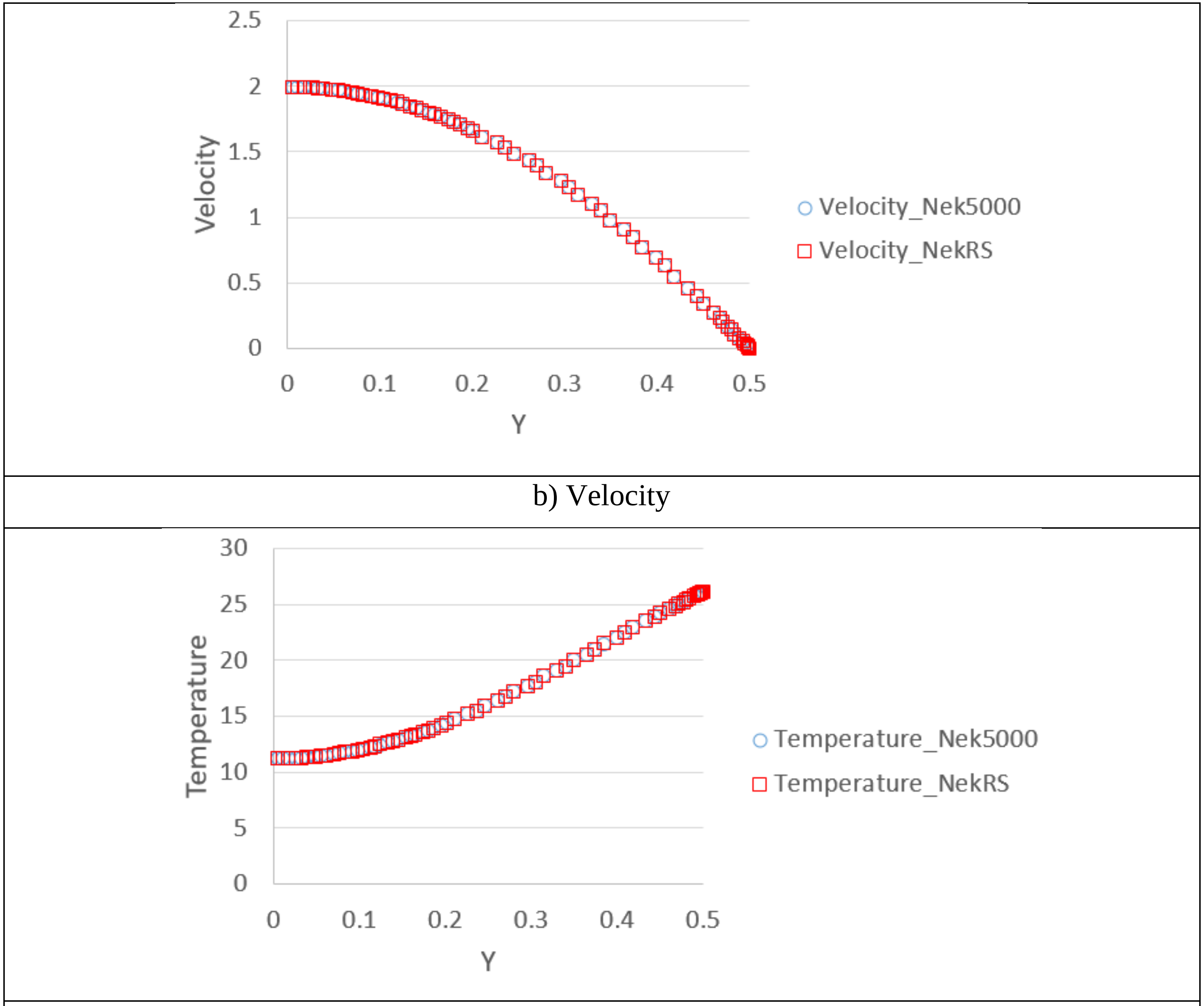
Flow in a tube is the simplest case for most industry applications. The purpose of this application is to test the basic workflow of NekRS, including mesh generation, boundary assignment, and output processing. The geometry of the domain is non-dimensionalized with tube diameter 1 and length 100. Lead flow was chosen so that the flow properties are the similar to that of LFR. The inlet velocity is set as a non-dimensional value 1. The Reynolds number of the tube case is chosen to be 500 and 5000 respectively. Besides testing the basic workflow of NekRS, the turbulent flow with Reynolds number 5000 in tube will also play a role in testing the averaging approach and Large Eddy Simulation (LES) solver in NekRS.

The basic workflow of NekRS was tested through building the case of laminar flow in tube (Re=500). The mesh is generated in ANSYS_ICEM and converted to the format that NekRS adapts, which indicates that the mesh can be generated from third party commercial tools or open source mesh tool kits. The "recycling" boundary condition is used for the inlet so that the fully developed velocity profile can be implemented at the inlet boundary (Figure 6.1a). Use of the recycling boundary condition in NekRS can reduce computational cost during HCF calculations. However, it is not as flexible as the recycling boundary condition in Nek5000. The mesh for NekRS needs to be generated by extruding a 2D mesh, which will constrain the ability to mesh some complex cases for HCF calculation. The results shown in Figure 6.1 confirm that the basic workflow of NekRS works well. The velocity and temperature distributions from selected locations are identical.

The averaging approach and LES solver in NekRS are tested through building the case of turbulent flow in tube (Re=5000). Since the transient turbulence flow in tube is chaotic (Figure 6.2), the results need to be averaged during some time. As shown in Figure 6.3, the contours of averaged velocity and temperature are symmetric and smooth through proper averaging process. In order to make a comprehensive qualitative comparison, the velocity and temperature contour from somewhere near outlet (z=80) and the center cross section(X=0) are extracted. Some quantitative comparison (Figure 6.4) are also made to verify the LES solver in NekRS. Good agreement is observed between NekRS and Nek5000 qualitatively and quantitatively. Since some numerical schemes are not yet available in NekRS yet, it is necessary to make sure all the numerical schemes, such as constant time step and filter weight, used in NekRS and Nek5000 are identical. Averaging the results in same and sufficiently long period of time is also essential to making a fair comparison



a) Velocity distribution at inlet boundary



b) Velocity

c) Temperature

Figure 6.1. Velocity and temperature comparison between NekRS and Nek5000 on laminar flow in tube with Reynolds number 500

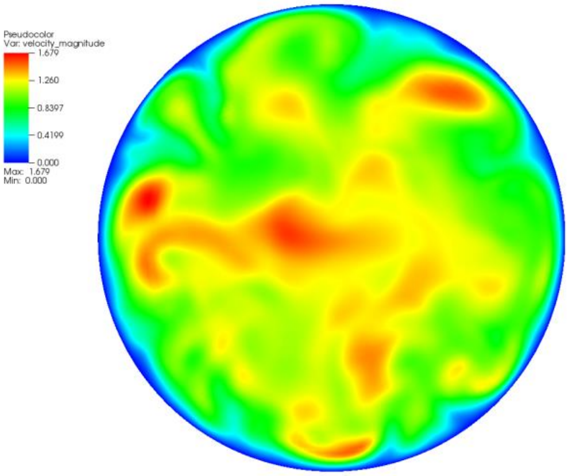
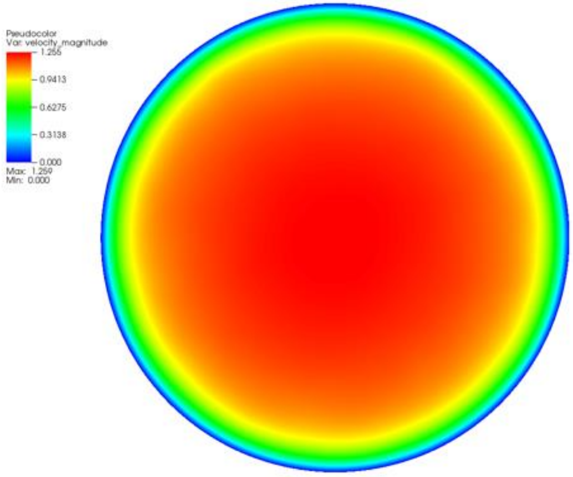
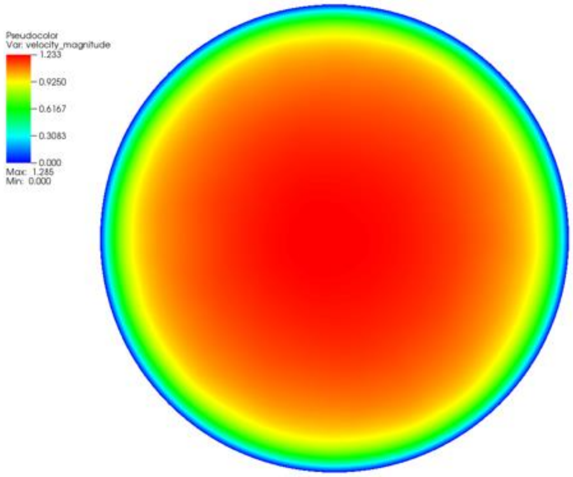
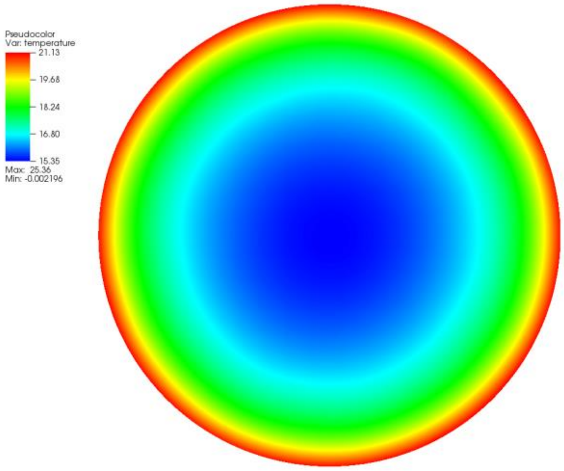
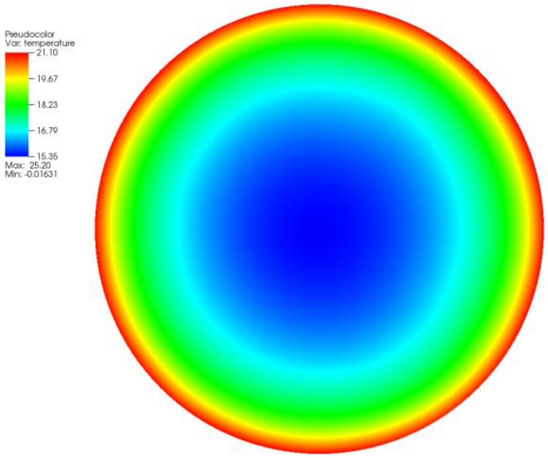


Figure 6.2. Transient snapshot of turbulent flow in tube with Reynold number 5000

	
NekRS velocity contour at Z=80	Nek5000 velocity contour at Z=80
	
NekRS temperature contour at Z=80	Nek5000 temperature contour at Z=80

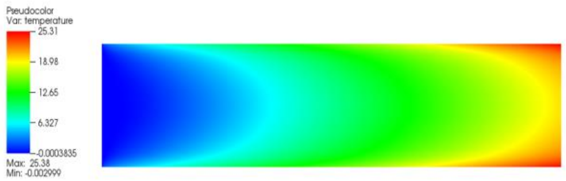
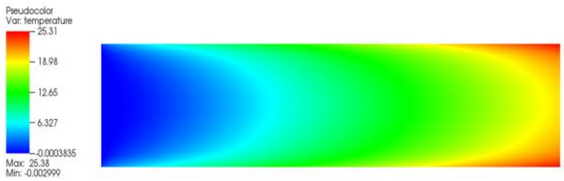
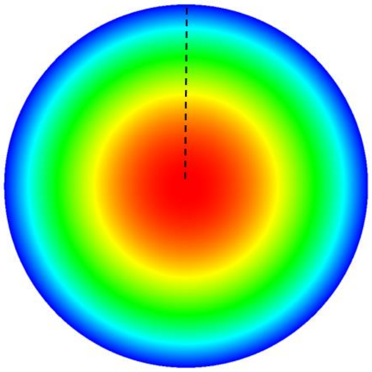
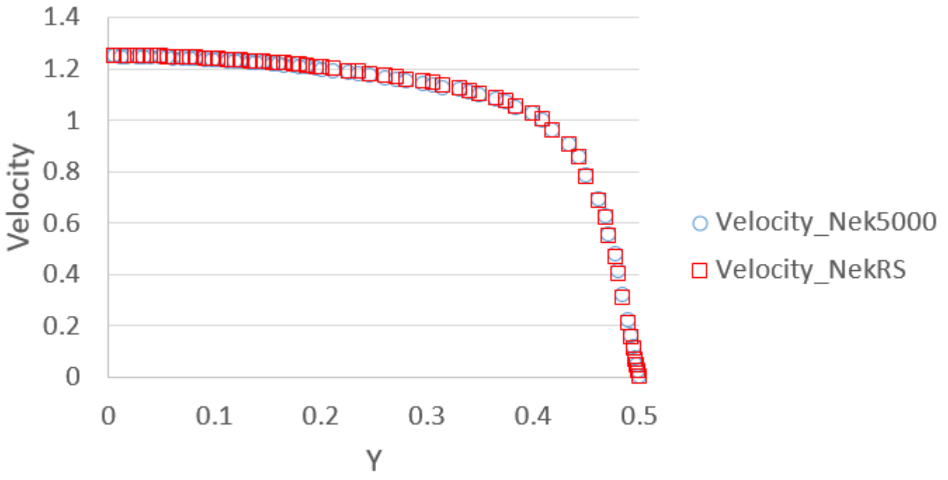
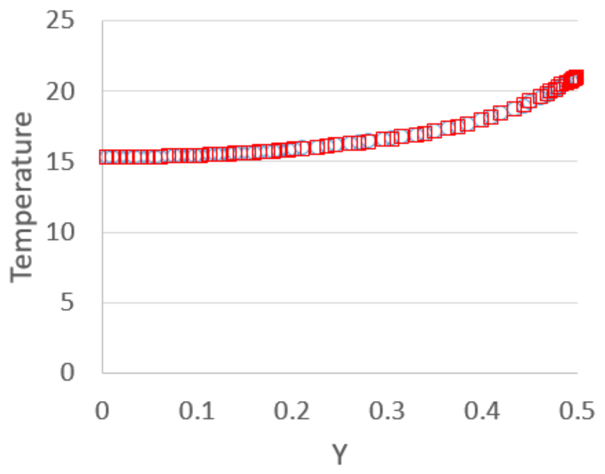
	
NekRS temperature contour at X=0	Nek5000 temperature contour at X=0

Figure 6.3. Contour of averaged velocity and temperature at different locations for turbulent flow in tube with Reynolds number 5000


a) Velocity distribuiton at inlet boundary

b) Velocity



c) Temperature

Figure 6.4. Velocity and temperature comparison between NekRS and Nek5000 on turbulent flow in tube with Reynolds number 5000

6.3 URANS Simulation on a Tube

LES simulation is usually too computationally expensive for engineering applications, especially for high Reynolds number flow. Reynolds-Averaged Navier-Stokes (RANS) model is more practical for most HCF calculations. The RANS equations are derived from the conservation of mass, momentum, and energy equations by expressing each term in the equation as the sum of a mean and a fluctuation. Because NekRS is based on the incompressible flow model, all such averages (even for the energy equation) are based on the notion of Reynolds averaging.

NekRS uses the k-τ turbulence model to close the mean flow equations [24]https://nekrsdoc.readthedocs.io/en/latest/references.html - kalitzin. Because the k-ϵ, k-ω, and k-ω SST models tend to dominate the RANS space, extra discussion is devoted here to motivating the use of this particular model. Because Pr_t is typically taken as a constant, often 0.90, the objective of incompressible flow RANS models is to compute the eddy viscosity and k needed to close the mean momentum and energy equations.

The k-τ model is a modification of the standard k-ω turbulence model that bases the second transport equation on the inverse of the specific dissipation rate ω, rather than on the ω. The k-τ model attempts to retain two important features of the k-ω model:

$$\tau = \frac{1}{\omega}$$

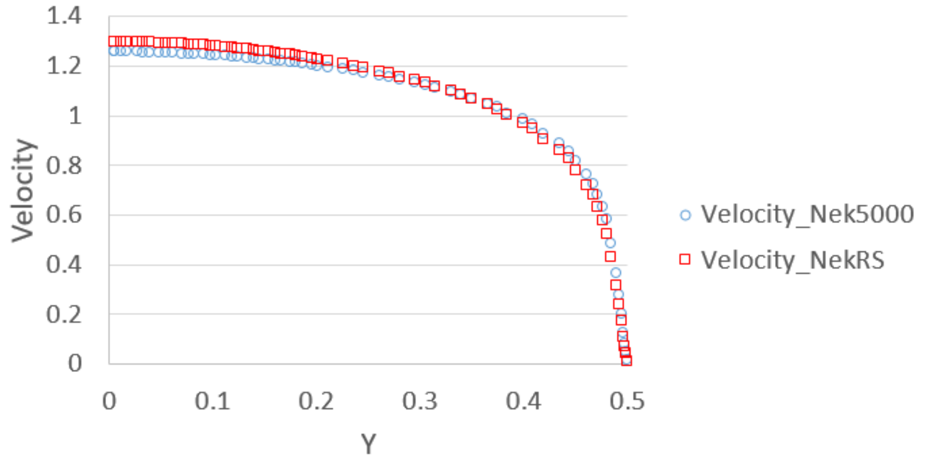
1. Good predictions for flows with adverse pressure gradients and separation, and
2. Reasonable prediction of boundary layers and near-wall behavior without wall functions or special low-Re_t treatments.

These two aspects contribute to better predictions of complex flows with reduced numerical complexity associated with wall functions or damping functions that can cause stiff behavior [24] and inaccurate flow predictions. By introducing the definition of τ=1/ω, the k-τ model attempts to improve upon the k-ω model in two main ways:

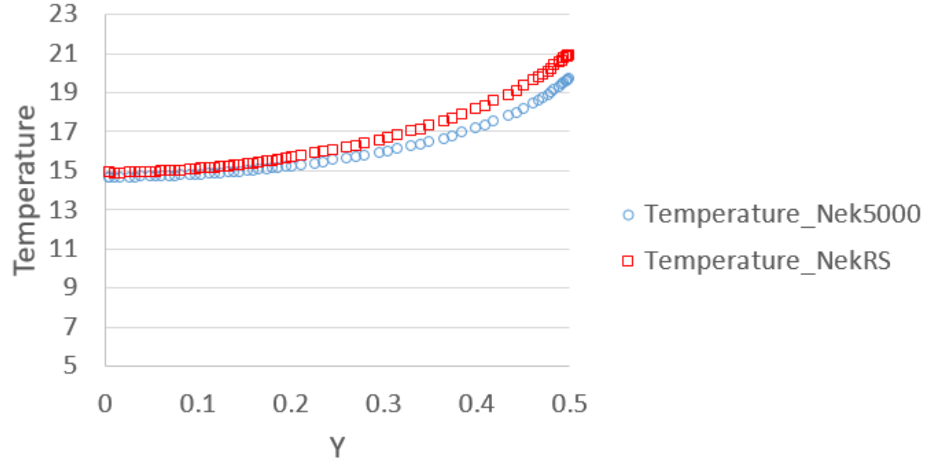
1. Simplify wall boundary conditions for the second transport equation.
2. Bound the source terms in the second transport equation in near-wall regions.

In this application, same geometry and fluid properties as described in Section 6.2 are used except for larger Reynolds number (Re=10000). Figure 6.5 shows the comparison of velocity, temperature, k and τ between NekRS and Nek5000 at the location shown in Figure 6.1a. Some difference is observed between the results from two codes. The deviation of k and τ in the open area suggests an issue in the k-τ model implementation in NekRS for dealing with the passive scalars of k and τ in the open area. Since the difference is small, it is hard to notice without comparing them one to one since the URANS simulation from NekRS looks qualitatively reasonable.

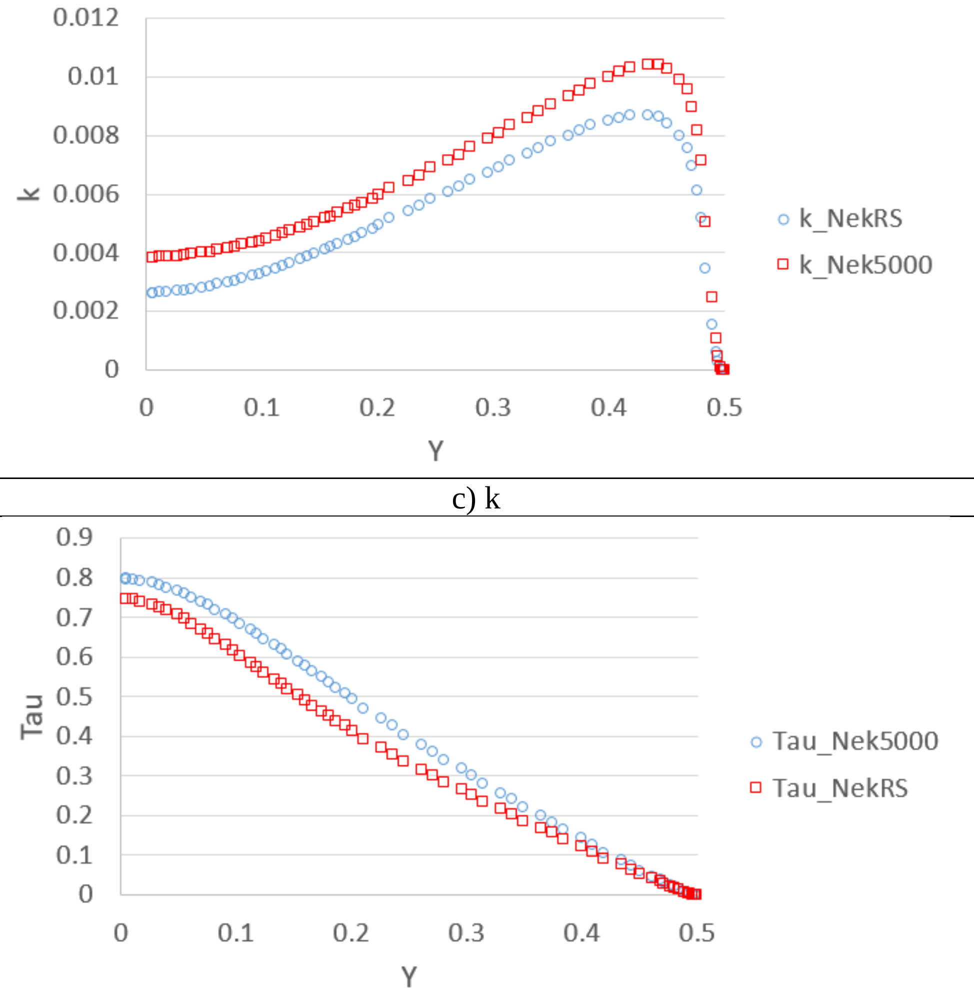
In addition to the difference observed in the results, the k-τ model in NekRS is less robust. It is necessary to set up a proper initial condition and boundary condition of k and τ to assure that the calculation does not diverge. In order to make a fair compassion of the URANS simulation capability in NekRS and Nek5000, the boundary condition of k and τ at the inlet boundary are identical.



a) Velocity



b) Temperature



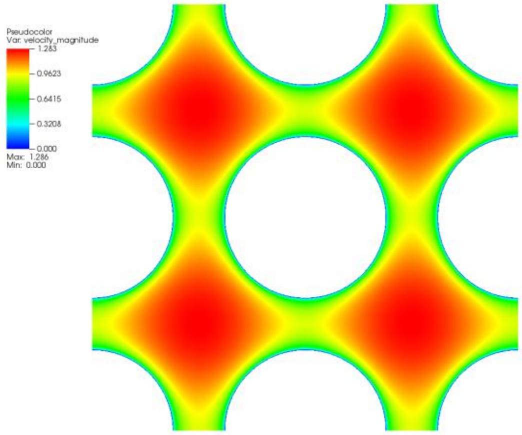
c) k

d) τ

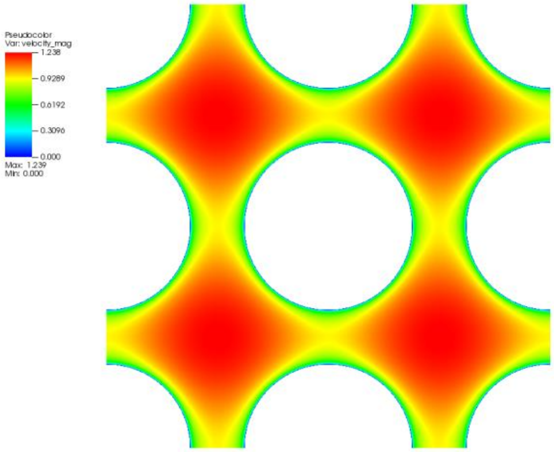
Figure 6.5. Comparison between NekRS and Nek5000 on turbulent flow in tube with Reynold number 10000

6.4 URANS Simulation on 2x2 Rod Bundle

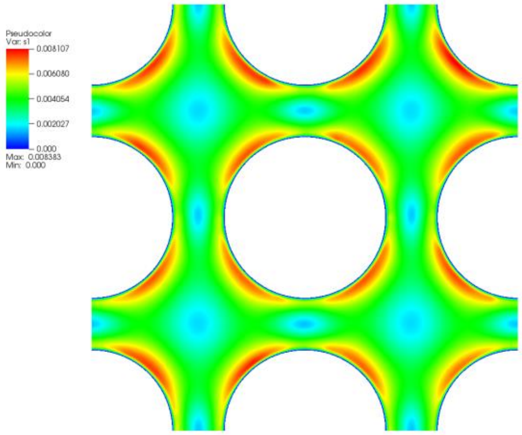
In order to further confirm the difference of URANS simulation between NekRS and Nek5000, iso-thermal URANS simulations on a 2x2 rod bundle in square array are performed to help to screen the issue. If the difference still exists, the source likely comes from k-τ itself in NekRS. The rod diameter is 1 and the length of the domain is 100 with Reynolds number 10000. Similar to the URANS simulation for tube case, the boundary condition of k and τ at inlet are kept identical in both NekRS and Nek5000 although it should not affect the results when the flow is fully developed. Figure 6.6 shows the contour of velocity, k and τ near outlet. Although each contour shows qualitatively reasonable distribution, some differences are observed especially on the contour of k and τ in the open area. Figure 6.7 shows the detailed distribution of velocity, k and τ at certain location, which further confirms the observation from the contour. The URANS simulation on the 2x2 rod bundle further confirms the findings in URANS simulation on the tube case. The k-τ model in NekRS need to be checked carefully to be used for HCF calculations for high Reynolds number flow such as that in the LFR.



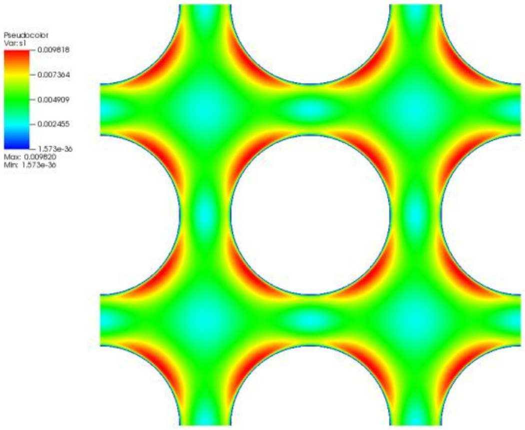
NekRS velocity contour at Z=80



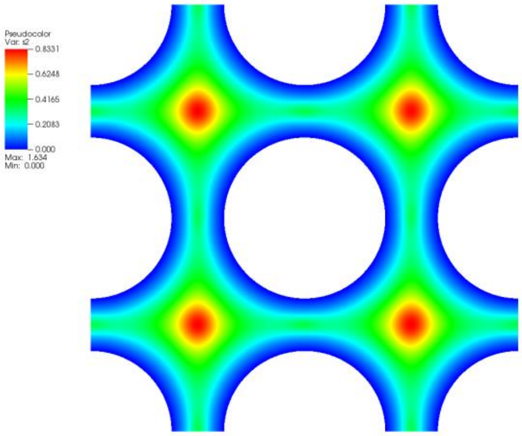
Nek5000 velocity contour at Z=80



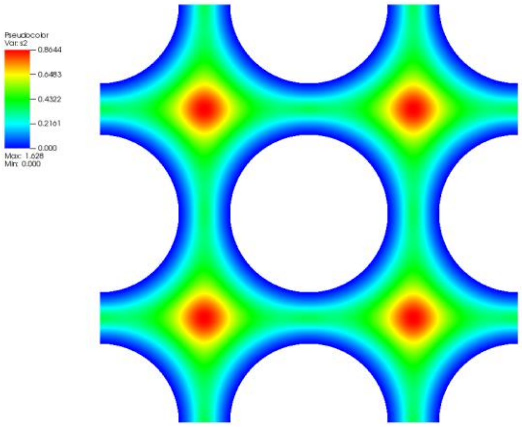
NekRS k contour at Z=80



Nek5000 k contour at Z=80

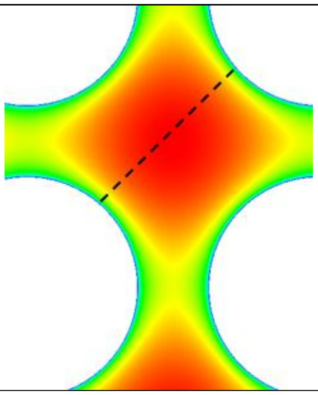


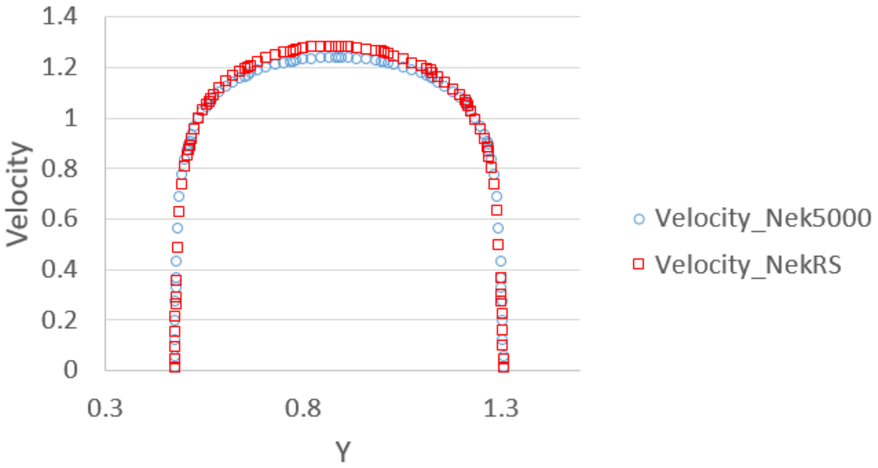
NekRS τ contour at Z=80



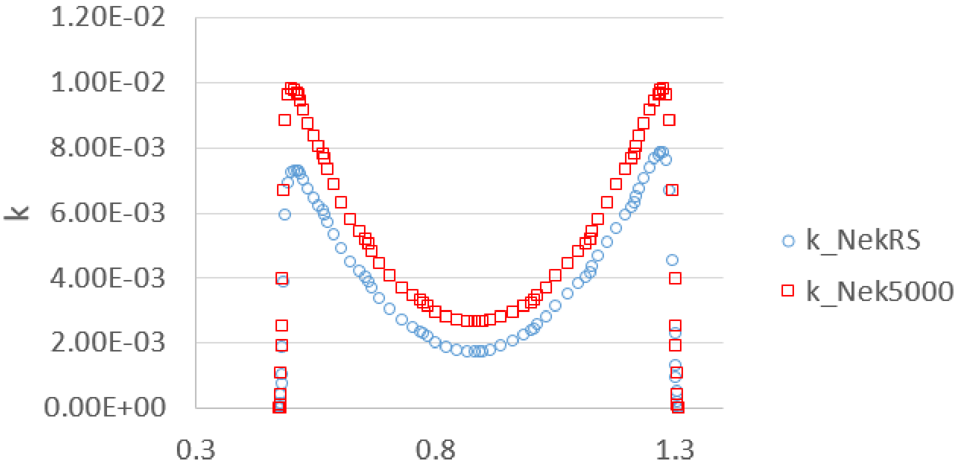
Nek5000 τ contour at Z=80

Figure 6.6. Contour of velocity, k and τ near outlet for turbulent flow in tube with Reynolds number 10000

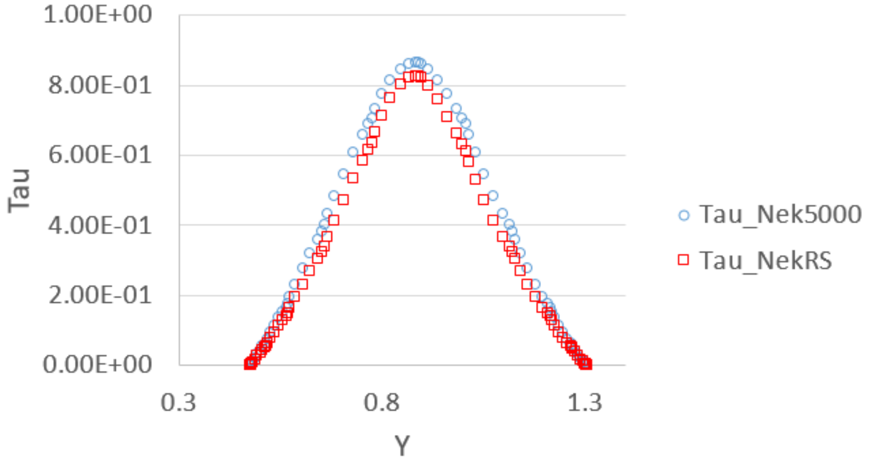




a) Velocity



b) k



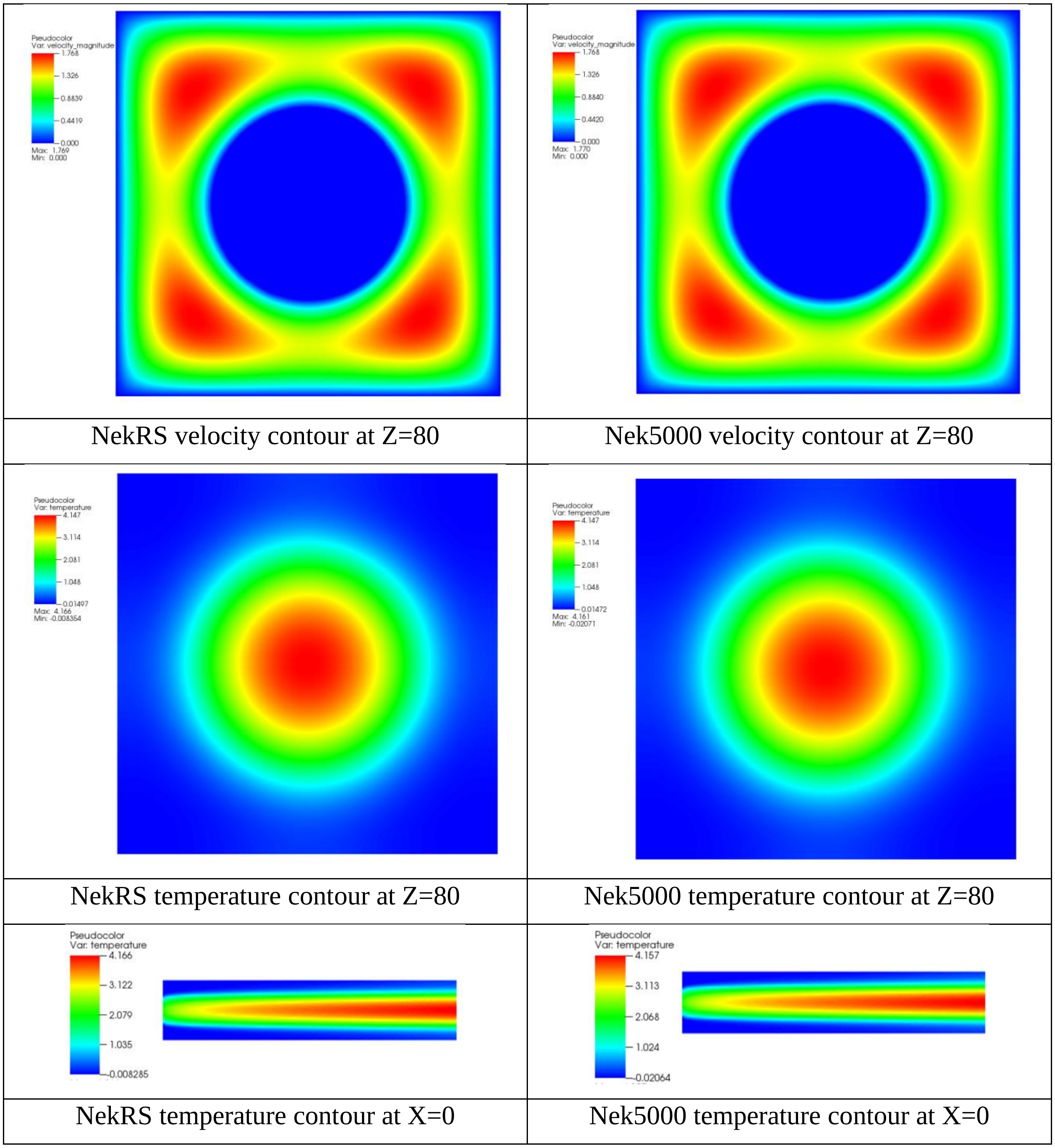
c) τ

Figure 6.7. Comparison between NekRS and Nek5000 on turbulent flow in 2x2 bundle with Reynold number 10000

6.5 LES Simulation on Pin Cell with Conjugate Heat Transfer

Conjugate heat transfer (CHT) is a very important module for HCF calculation since several HCFs originate from uncertainties in the solid region. However, the setup for CHT in NekRS is different from that in Nek5000. It is necessary to verify the essential capability needed for CHT calculation in NekRS. The mesh for conjugate heat transfer includes the meshes for multiple solid and fluid regions. The purpose of this application is to test the basic workflow for CHT setup in NekRS including the mesh generation, mesh conversion, boundary assignment and variable properties for multiple domains. The application is simplified to a pin cell in a duct. The diameter of the pin is 1 and the edge of the duct is 2 with Reynolds number 780.

The results from NekRS and Nek5000 are compared near the outlet and at the center cross section (Figure 6.8). The results agree well with each other qualitatively. The velocity in the solid domain is 0. The solid domain is heated up along the stream wise direction while the center of the pin reaches the maximum temperature. The detailed comparisons at specific location (Figure 6.9) also verify the results from NekRS.



NekRS velocity contour at Z=80

Nek5000 velocity contour at Z=80

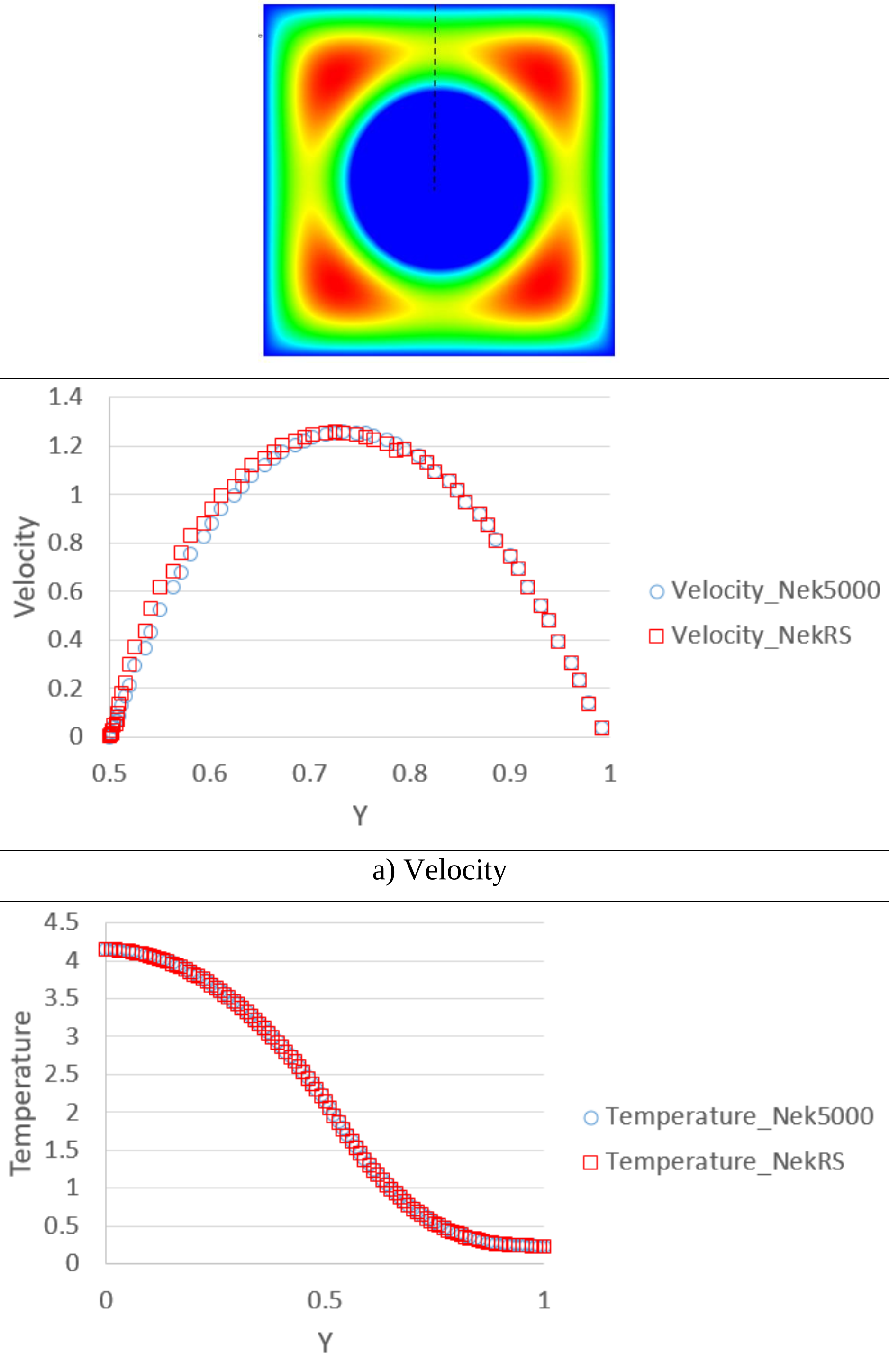
NekRS temperature contour at Z=80

Nek5000 temperature contour at Z=80

NekRS temperature contour at X=0

Nek5000 temperature contour at X=0

Figure 6.8. Contour of velocity and temperature at different locations for laminar flow (Reynold number 780) in pin cell with CHT



a) Velocity

b) Temperature

Figure 6.9. Velocity and temperature comparison between NekRS and Nek5000 on (Reynold number 780) in pin cell with CHT

6.6 LES Simulation on 7-Pin Rod Bundle with Conjugate Heat Transfer

A demonstration 7-pin rod bundle model was developed which consists of a single hexagonal assembly containing 7 cladded fuel pins. The mesh configuration of the model is shown in Figure 6.10. The green part is the fluid region while the light blue is the duct. All the pins contain the solid domain for both cladding and fuel pellet. The detailed specification of the 7-pin rod bundle is listed in Table 6.1, which comes from the WEC LFR design. The purpose of this application is to verify the capabilities in NekRS, including assigning variable properties for solid domains and assigning variable heat sources for different solid domains. To simplify the problem, all the solid domains use the same properties.

Figure 6.11 and Figure 6.12 verify that NekRS can reproduce identical results for the 7-pin rod bundle with conjugate heat transfer as Nek5000 does. The variable heat source can be assigned to fuel pellet domain properly, which is crucial for the coupled calculation on HCF.

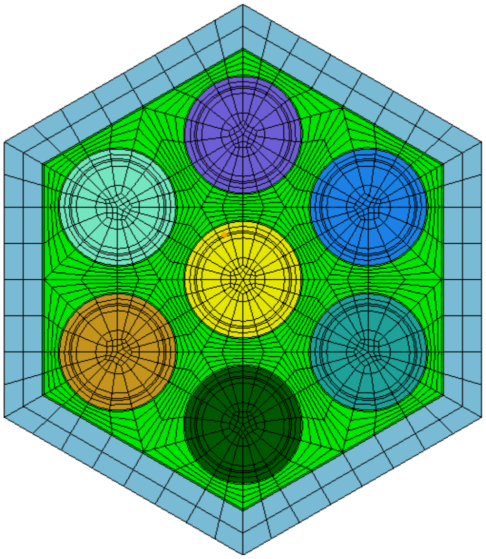
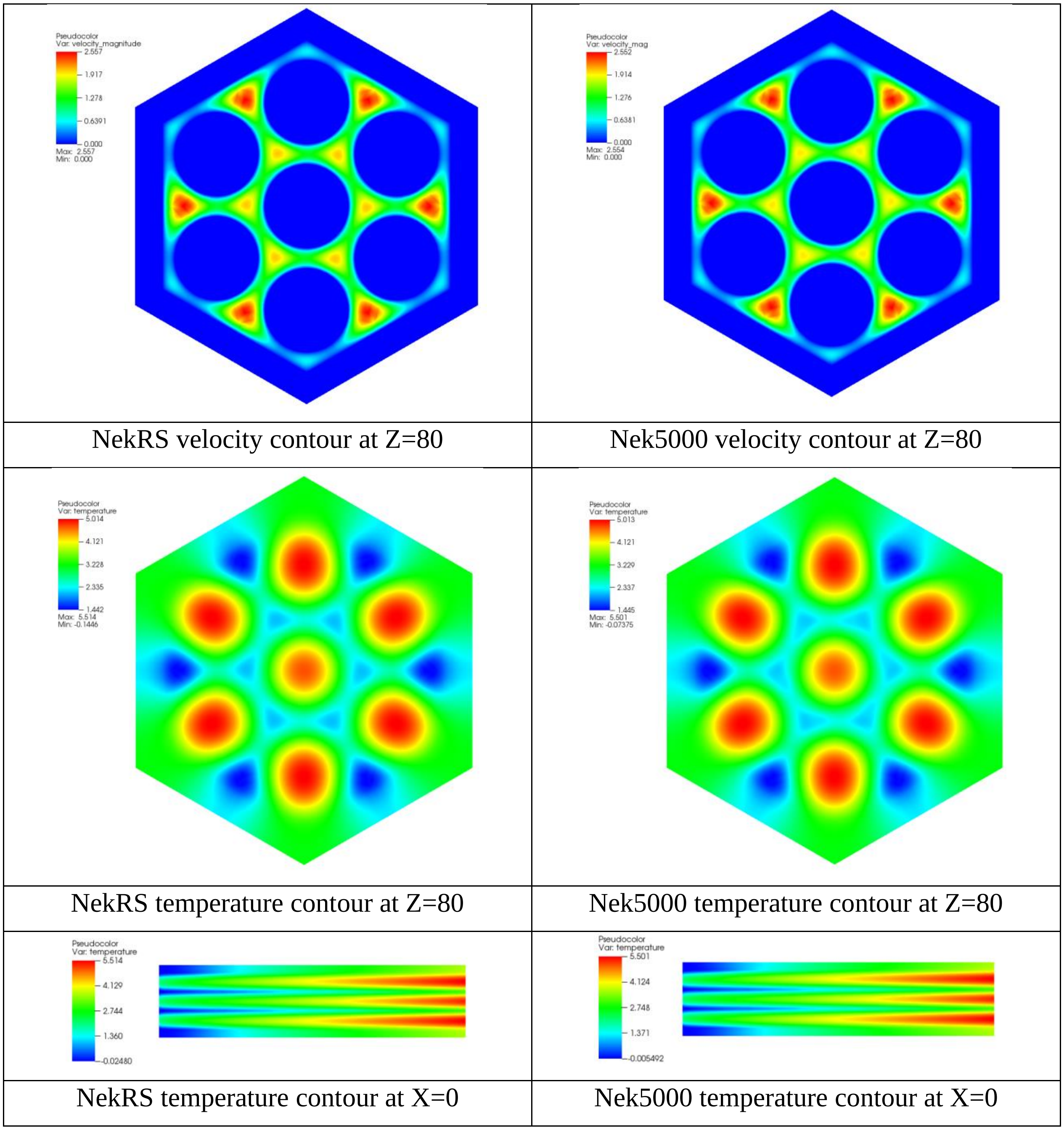


Figure 6.10. Mesh configuration of 7-pin rod bundle with CHT

Table 6.1. Specification of 7-pin rod bundle model

Parameter	Dimensional	Dimensionless
cladding outer diameter (cm)	1.08	1.00
cladding inner diameter (cm）	0.90	0.83
cladding thickness (cm)	0.11	0.10
pin pitch （cm）	1.34	1.24
P/D	1.24	1.24
inner duct wall flat-to-flat distance (cm)	15.34	14.20
outer duct wall flat-to-flat distance (cm)	16.05	14.85
duct wall thickness （cm）	0.35	0.33
fuel pellet outer diameter (cm)	0.86	0.80
active core height (cm)	106.07	98.15
coolant density (kg/m3)	10401.75	1
coolant viscosity (Pa-s)	0.171	1/780
coolant specific heat (J/kg-K)	144.19	1
coolant thermal conductivity (W/mK)	18.09	1/1063
ρCp	1499828.333	1
cladding density (kg/m3)	7970	1
cladding thermal conductivity (W/mK)	21.6	$1/1063\times k_{solid}/k_{fluid}$
cladding specific heat (J/kg K)	580	$\rho Cp_{solid}/\ \rho Cp_{fluid}$
ρCp	4622600	ρCpsolid/ ρCpfluid
inlet velocity (m/s)	1.186	1
inlet temerature (C)	420	0
ΔT	100	100
characteristic length (cm)	1.081	1
volumetric heat flux (W/m3)	6.00E+08	0.036
Reynolds number	780	780
Prandtl number	1.363	1.363
Peclet number	1063	1063



NekRS velocity contour at Z=80

Nek5000 velocity contour at Z=80

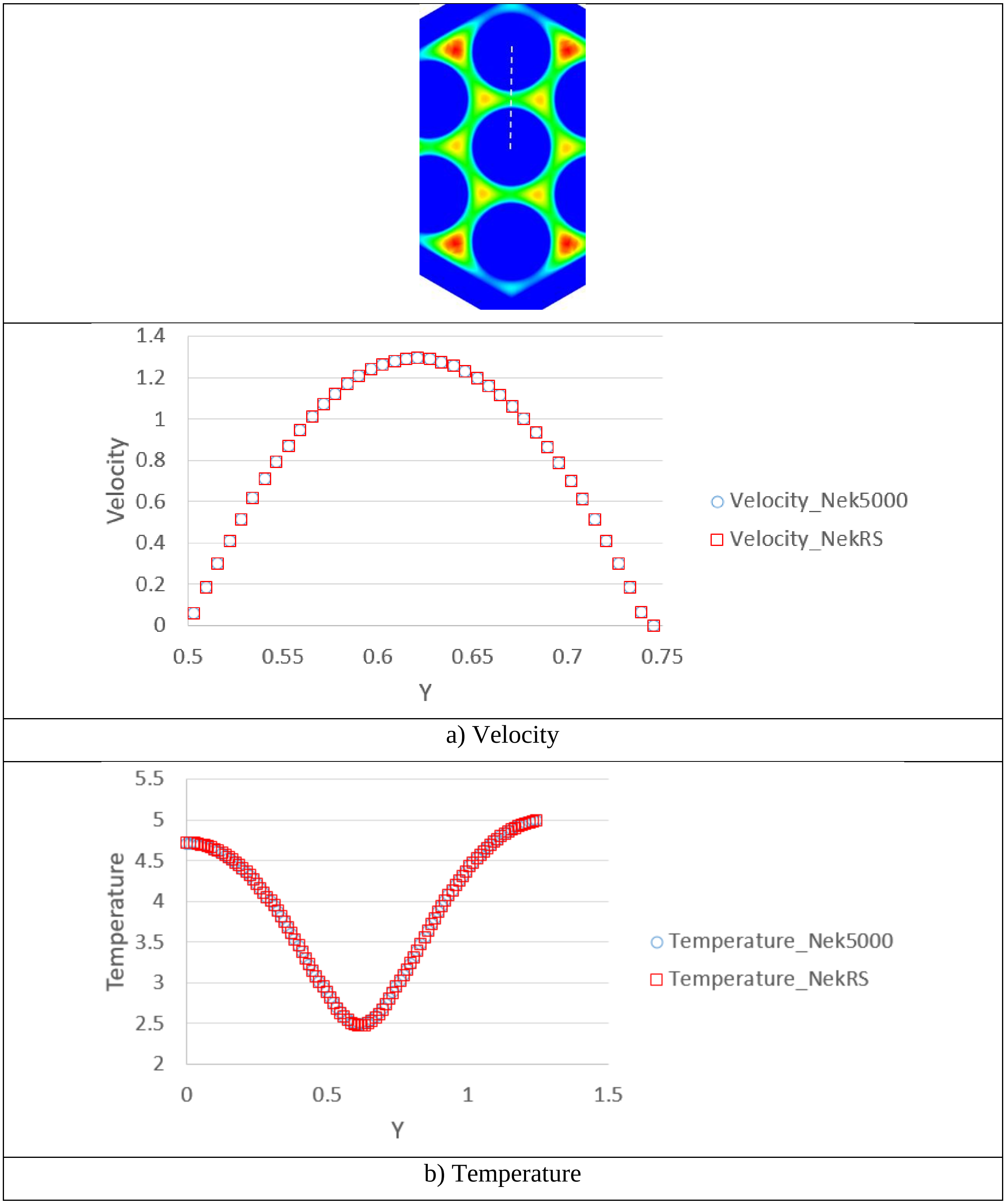
NekRS temperature contour at Z=80

Nek5000 temperature contour at Z=80

NekRS temperature contour at X=0

Nek5000 temperature contour at X=0

Figure 6.11. Contour of velocity and temperature at different locations for laminar flow (Reynold number 780) in 7 pin bundle with CHT



a) Velocity

b) Temperature

Figure 6.12. Velocity and temperature comparison between NekRS and Nek5000 on (Reynold number 780) in 7 pin bundle with CHT

6.7 Summary

This project used multiple applications to test the capabilities in NekRS needed for HCF calculation. The results from Nek5000 are used as references to compare against. The summary of the applications tested is listed in Table 6.2. The LES simulation on laminar flow in tube is used to test the basic workflow in NekRS including mesh generation, mesh conversion and boundary assignment. This application also takes advantage of the recycling boundary condition to save computational cost. However, the recycling boundary condition implemented in NekRS is not as flexible as that in Nek5000. It can be only applied to a mesh that is generated from 2D to 3D extrusion. The LES simulation on turbulent flow in tube is used to test the LES solver in NekRS. Since the turbulent flow in chaotic in transient state, the application is also used to test the averaging approach in NekRS to get accurate statistic results. The RANS simulation on turbulent flow in both tube and 2x2 rod bundle confirms that there exist some differences between NekRS results and Nek5000 results. The k-τ model in NekRS needs to be checked carefully before it can be used for further HCF calculation. In addition, the k-τ model in NekRS is less robust. A more generic routine for setting up boundary condition of k and τ in arbitrary geometry is preferred. The LES simulation for pin cell with conjugate heat transfer is used to test the basic workflow for CHT setup including the mesh generation, mesh conversion, boundary assignment and variable properties for multiple domains. The recycling boundary cannot be used for CHT case. The LES simulation for 7pin rod bundle with conjugate heat transfer is used to verify the capabilities in NekRS of assigning variable heat source for different solid domain.

Table 6.2. Summary of applications tested for NekRS

Application	Simulation Type	Flow	Purpose of the application	Features to be improved
Tube	LES	laminar	○ Test the basic work flow ○ Test recycling boundary condition	○ Flexibility of recycling boundary
Tube	LES	turbulent	○ Test LES solver ○ Test averaging approach	
Tube	URANS	turbulent	○ Test RANS model in NekRS	○ Check k-τ model in NekRS ○ Robustness in RANS model
2x2 rod bundle	URANS	turbulent	○ Confirm the issue found in tube case	
Pin cell	LES+CHT	laminar	○ Basic workflow for CHT setup	○ Usability of recycling boundary
7-pin rod bundle	LES+CHT	laminar	○ Assign property and hear source for different solid region	○ Flexibility of using number of Boundary ID

In summary, NekRS has the basic capabilities needed for HCF calculation. Some features in NekRS need to be improved to make it mature enough for HCF calculation.

7 Overview of MOOSE-Wrapped NekRS (Cardinal App)

Full scale HCF calculations with Griffin and NekRS will be the focus of future work, pending readiness and computational efficiency of both codes. This section provides a brief introduction to the Cardinal app (MOOSE-wrapped NekRS) which will be used in the future.

Cardinal is a MOOSE-based multiphysics application consisting of wrapping of the GPU-based spectral element Computational Fluid Dynamics (CFD) code NekRS and the Monte Carlo particle transport code OpenMC [25] within the MOOSE framework. Cardinal is intended for providing high-resolution thermal-hydraulics and/or nuclear heating feedback to MOOSE multiphysics simulations. Cardinal has capabilities for:

- Conjugate Heat Transfer (CHT) coupling to NekRS
- Volumetric source term coupling to NekRS
- Volumetric source term, density, and temperature coupling to OpenMC

All three of the above features can be combined together for complex calculations. A geometry-agnostic and MOOSE-application-agnostic coupling allows NekRS and OpenMC to be coupled to any MOOSE application, enabling a broad set of multiphysics capabilities. Advantageous features of Cardinal's design include:

- In-memory coupling, eliminating the need to write code-specific I/O programs and reducing potential file-based communication bottlenecks
- Distributed parallel meshes for very large-scale problems
- Automatic construction of geometric mappings between the NekRS CFD mesh, the OpenMC geometry, and the coupled MOOSE meshes, with virtually no requirements on node/element/cell alignment. This eliminates the need for rigid one-to-one mappings and enables multiphysics calculations on arbitrary geometries.
- Straightforward multiphysics problem setup

The coupling capability from NekRS and OpenMC will provide the basis for future NekRS and Griffin coupling.

8 Early Exploration of Stochastic Tools for Perturbed Conditions

In earlier chapters we concentrated on Griffin's capability to solve the nominal condition problem. For hot channel factor determinations, it will be necessary to simulate the 3D LFR assembly many times with perturbations in geometry and material composition. An advantage that Griffin has over other neutronics packages is that it provides access to the Stochastic Tools module native to the MOOSE framework. This package provides a large suite of tools for conducting stochastic analysis with physics-based MOOSE applications. Using these tools, it is possible to implement a complete parameter study with only two input files: one for the stochastic driver, and one for the physics, with uncertain parameters sampled and adjusted on-the-fly. It also provides access to more robust tools for sampling parameter domains, as well as for post-processing results collected from each perturbed case.

We have done some early exploration of the tools available in this package, and how a parameter study can be configured for the 3D LFR assembly. Some testing has been done with a working set of inputs, but early results are not yet available. The following sections will detail how a parameter study can be configured for one parameter of interest for hot channel factor determination – the variability in fissile content. There is also a discussion on limitations applying these tools in Griffin, and a summary of how we believe these tools will fit in to the larger project.

8.1 Input Setup via MOOSE's MultiApp System

The Stochastic Tools module uses the MultiApp structure, similar to the implementation of diffusion-based acceleration in Griffin. A main application is created to serve as a driver for the stochastic sampling process. The driver handles the sampling of random perturbations from one of the available random distributions, transferring perturbed parameters to the physics-based sub-application, and processing data for quantities of interest. A sample input syntax is shown in Figure 8.1, in which the input "fissmaldist_STM.i" serves as the driver, and "fissmaldist_DFEM_sn.i" is a sub-application for performing an unaccelerated DFEM-SN solve. This transport input is identical to the input in Figure 5.1, differing only in the material mixtures defined in MicroNeutronicsMaterial - therefore, only the relevant section of the materials block is shown in the sample.

In previous efforts determining hot channel factors using PROTEUS-MOC, the variability in plutonium-239 enrichment in the fuel was one of the uncertain parameters examined. In that example, the enrichment of individual fuel pins was perturbed, treating the enrichment as a Normal-distributed random variable with the nominal condition as the mean and a standard deviation of the Pu-239 enrichment of 1.67% (m=3.4133E-03 cm-3, s=5.7002E-05 cm-3).The input sample in Figure 8.1 is for a simplified case, in which the enrichment of entire rings of fuel pins, rather than each pin individually, is perturbed with the same distribution. This was a simpler case to implement for early exploration of MOOSE's stochastic options. In this sample, the parameters "r1_pu239" and "r1_pu240" in the transport input are used to define the density of each isotope in the first ring of fuel pins. When this sub-application is initialized, the driver will transfer a new value to r1_pu239 corresponding to a perturbed case. The command defining r1_pu240 will then adjust the plutonium-240 content to maintain overall constant density. Finally, these parameters are used in defining the fuel material composition in ring 1. This process is mimicked for all other rings. Each instance of this transport sub-app will correspond to a unique perturbed case generated on-the-fly by the stochastic driver application.

8.2 Stochastic Tools Inputs for Fissile Maldistribution Sampling

The Stochastic Tools Module uses additional input blocks that are non-standard to most MOOSE-based simulations. In the main input, the `StochasticTools` block is used to declare the application as a driver for a Stochastic Tools problem. The `Distributions` block is used to define random distributions from which samples can be drawn. In this case, a Normal random variable is defined with the nominal Pu-239 enrichment as the mean. This one Distribution object can be used to draw samples for all seven fuel rings.

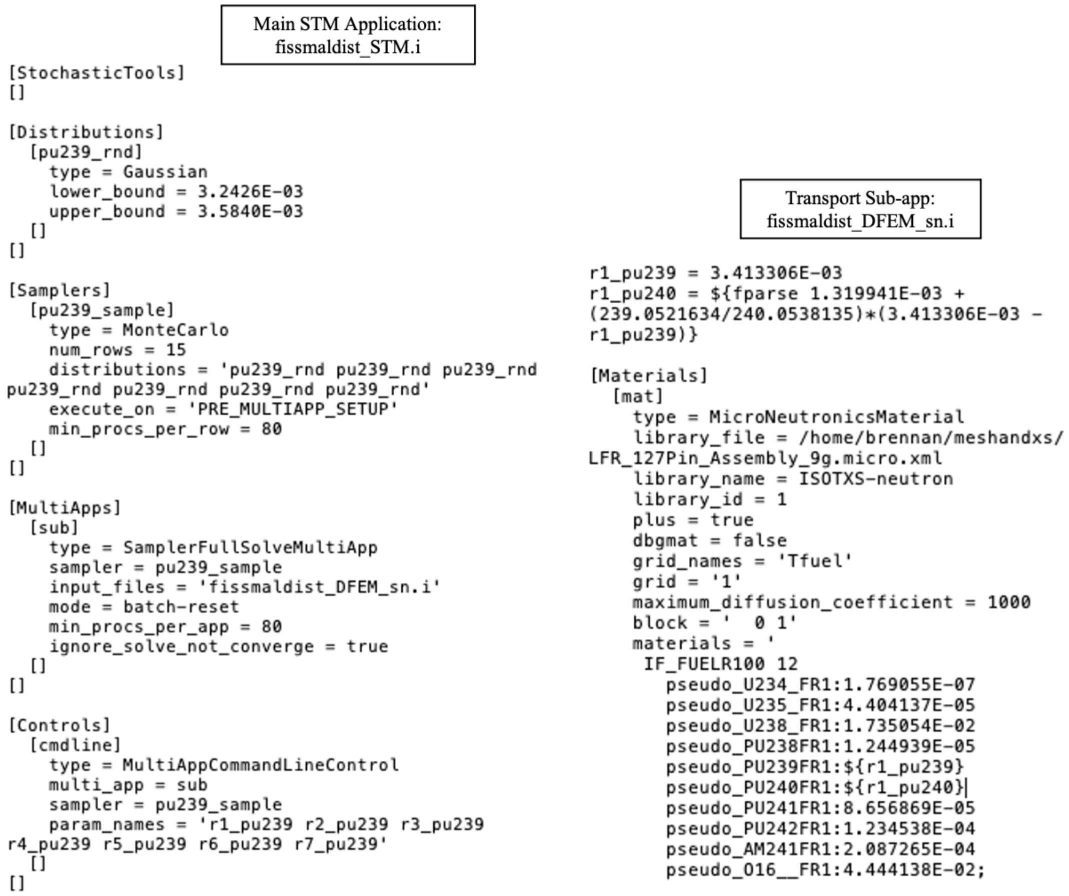
The `Samplers` block is used to define a process for generating samples from the available Distribution objects. In this case, `MonteCarlo` sampling is used, but the Stochastic Tools Module supports other standard sampling techniques - most notably, it supports Latin Hypercube sampling. This is a well-known technique for improving the range of parameter perturbations covered by a fixed number of random samples and should be considered when using these options for HCF determination in the future. The `num_rows` option is used to set the number of perturbed cases to generate (set to 15 in this example). For each row (case), samples are drawn according to the space-delimited array defined in the Samplers/distributions input. In Figure 8.1, each row will draw seven independent samples from the "pu239_rnd" distribution. This will correspond to randomized enrichments for each ring of fuel pins, such that each row in the sampler matrix corresponds to an independent perturbed case. The execute-on option specifies when these samples should be generated - because we are creating independent eigenvalue problems, these can be generated prior to setup of the transport application.

The `min_procs_per_app` option is used to set what configuration should be used for the parallelism of the problem. By default, the stochastic tools driver will attempt to split all available cores evenly across all rows in the sampler matrix, such that each perturbed case is run in parallel on a small number of cores each. `min_procs_per_app` can be used to prevent the driver from under-allocating cores for any given task. If there are not enough cores to meet this minimum for all cases simultaneously, the driver will wait for problems to finish before running the next case, thereby serializing the simulations. This option is necessary for the 3D LFR single-assembly problem, as many cores are typically needed to meet the runtime and memory requirements. On a shared, SLURM-based job submission system such as Bebop, it is also typically more efficient to request a smaller number of cores and run for a longer time, than to request more cores to run in a short time. For this reason, it is recommended that `min_procs_per_app` be set to the number of cores available (so that the perturbations are run entirely one after another), or some sensible minimum is used if more computational resources are available.

In the `MultiApps` input block, a `SamplerFullSolveMultiApp` is declared. This causes the driver to create full-solve sub-applications for each perturbed case. This option should typically be used for hot channel factor calculations, which rely on independent eigenvalue problems. The `input_files` option provides a path to the input for the sub-application. Occasionally, there may be perturbed cases that do not converge. To prevent the application from exiting completely when this happens, the option `ignore_solve_not_converge` can be set. This causes the driver to skip over failed cases and report them to the user gracefully, rather than aborting the main process.

The `Controls` block is used to instruct the driver what parameters should be perturbed in the sub-application, and how that transfer should be executed. For certain parameter types, such as boundary conditions, a Control object with the type `SamplerReceiver` can be defined in the sub-application input. Then, a `Transfer` block must be defined in the main application with the

`SamplerParameterTransfer` type. Most uncertain parameters of interest in this hot channel factor study are incompatible with this configuration because they must be assigned prior to mesh generation or input parsing. In the case of the fissile content, the compositions in `MicroNeutronicsMaterial` must be altered using command line arguments. To accomplish this, a Controls object is defined in the main input with the `MultiAppCommandLineControl` type. The `multi_app` and sampler inputs define which sub-application to target, as well as which sampler matrix to use. The `param_names` option is then used to provide a space-delimited array of parameters in the sub-application to target for perturbation.



Main STM Application:
fissmaldist_STM.i

```
[StochasticTools]
[]

[Distributions]
  [pu239_rnd]
    type = Gaussian
    lower_bound = 3.2426E-03
    upper_bound = 3.5840E-03
  []
[]

[Samplers]
  [pu239_sample]
    type = MonteCarlo
    num_rows = 15
    distributions = 'pu239_rnd pu239_rnd pu239_rnd
pu239_rnd pu239_rnd pu239_rnd pu239_rnd'
    execute_on = 'PRE_MULTIAPP_SETUP'
    min_procs_per_row = 80
  []
[]

[MultiApps]
  [sub]
    type = SamplerFullSolveMultiApp
    sampler = pu239_sample
    input_files = 'fissmaldist_DFEM_sn.i'
    mode = batch-reset
    min_procs_per_app = 80
    ignore_solve_not_converge = true
  []
[]

[Controls]
  [cmdline]
    type = MultiAppCommandLineControl
    multi_app = sub
    sampler = pu239_sample
    param_names = 'r1_pu239 r2_pu239 r3_pu239
r4_pu239 r5_pu239 r6_pu239 r7_pu239'
  []
[]
```

Transport Sub-app:
fissmaldist_DFEM_sn.i

```
r1_pu239 = 3.413306E-03
r1_pu240 = ${fparse 1.319941E-03 +
(239.0521634/240.0538135)*(3.413306E-03 -
r1_pu239)}

[Materials]
  [mat]
    type = MicroNeutronicsMaterial
    library_file = /home/brennan/meshandxs/
LFR_127Pin_Assembly_9g.micro.xml
    library_name = ISOTXS-neutron
    library_id = 1
    plus = true
    dbgmat = false
    grid_names = 'Tfuel'
    grid = '1'
    maximum_diffusion_coefficient = 1000
    block = '  0 1'
    materials = '
     IF_FUELR100 12
       pseudo_U234_FR1:1.769055E-07
       pseudo_U235_FR1:4.404137E-05
       pseudo_U238_FR1:1.735054E-02
       pseudo_PU238FR1:1.244939E-05
       pseudo_PU239FR1:${r1_pu239}
       pseudo_PU240FR1:${r1_pu240}
       pseudo_PU241FR1:8.656869E-05
       pseudo_PU242FR1:1.234538E-04
       pseudo_AM241FR1:2.087265E-04
       pseudo_O16__FR1:4.444138E-02;
```

Figure 8.1. Sample input for implementing a parameter study for fissile content maldistribution. Left: Input for main, Stochastic Tools application. Right: Input for transport sub-application with parameters adjusted by the main driver.

8.3 Compatibility with Griffin

Currently, it is not possible to run a Stochastic Tools-driven parameter study with Griffin's implementation of NDA for DFEM-SN. This is because both the Stochastic Tools Module and the NDA implementation in Griffin make use of MOOSE's MultiApp framework. The `MultiAppCommandLineControl` option can only transfer values to parameters defined in the sub-application directly nested under the Stochastic Tools driver. Figure 8.2 shows a side-by-side comparison of the MultiApp structure for implementing a parameter study with NDA in Griffin (left), as well as with unaccelerated DFEM-SN or CMFD(right). In this example, the driver spawns two sub-applications simultaneously to run independent perturbed cases. To do this with Griffin's NDA, it calls the diffusion application, which then spawns transport sub-applications to handle sweep updates. The driver only has access to parameters defined by the diffusion input, which is, for the most part, only used to define executioner and solver options for the diffusion problem. Most parameters of interest are defined in the transport application, such as material compositions and mesh geometry (if using an internal MOOSE-generated mesh). To perform a parameter study with NDA, inputs would have to be hard-coded as they have been in prior work. The Stochastic Tools options are, however, compatible with unaccelerated DFEM-SN. In this case, the transport application is nested directly under the stochastic driver, such that parameters of interest are accessible.

The recent implementation of CMFD acceleration in Griffin does not use the MultiApp structure, and relies on a single input file. Because of this, it is possible to use CMFD to accelerate perturbed cases when used with the Stochastic Tools driver. The MultiApp structure is identical to that of unaccelerated DFEM-SN when using CMFD, as reflected in the right flow-chart in figure 8.2. Though performance improvements using CMFD have not yet been assessed for the nominal condition problem, this is another option that should be explored for HCF determination.

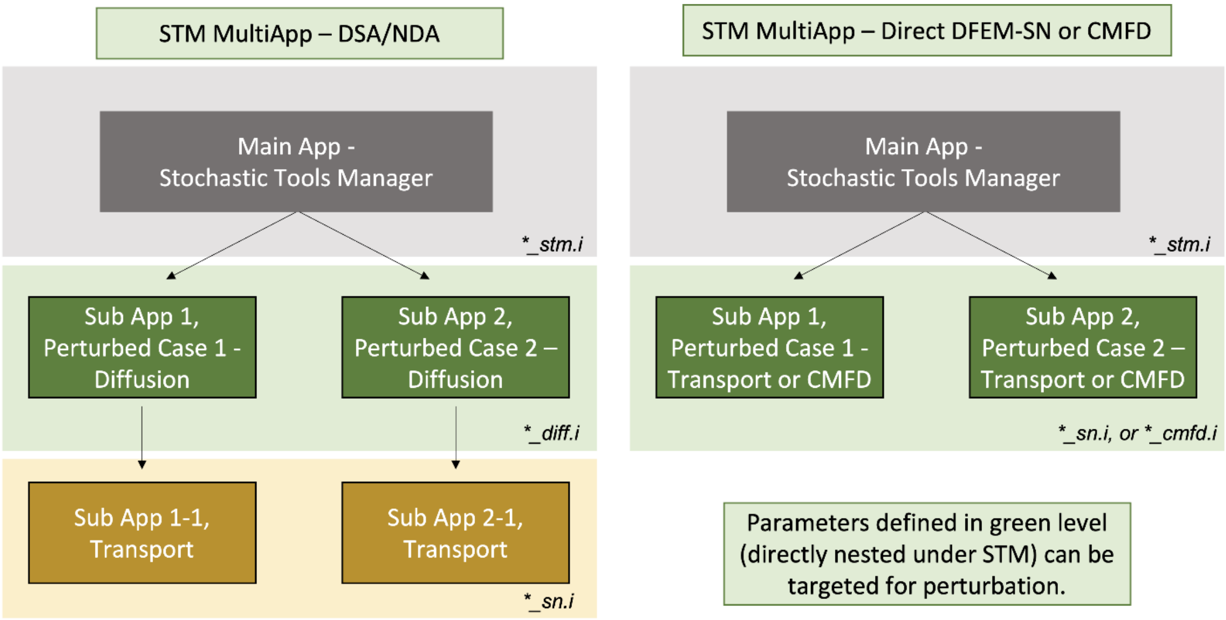


Figure 8.2. Diagram of MultiApp structure for driving a parameter study with MOOSE's Stochastic Tools in Griffin. Left: MultiApp for a DSA/NDA simulation. Right: MultiApp for an unaccelerated DFEM-SN simulation.

For this reason, the syntax example in Figure 8.1 is implemented with unaccelerated DFEM-SN. However, as shown in the performance study in Section 5.5, coarse-mesh diffusion acceleration was very effective for reducing runtime and memory consumption for the nominal condition problem, and would likely be an appropriate choice for simulating perturbed cases. If future Griffin updates allow for transferring perturbed parameters to further-nested sub-applications, it would be possible to take advantage of the performance improvements from NDA while using the Stochastic Tools Module for determination of hot channel factors. However, this may be unnecessary with the new implementation of CMFD in Griffin, as it uses a single application and should be compatible with the Stochastic Tools driver.

8.4 Compatibility with New Mesh Generators

As detailed in Section 4.3.2, all work so far in Griffin has made use of the Argonne Mesh Tools for mesh preparation. Because the meshes are generated externally to MOOSE, it is not possible to use the Stochastic Tools options to conduct a geometry-focused parameter study. This presents an issue, as cladding thickness has been identified as an important uncertainty for hot channel factor studies. Fortunately, enhancements to MOOSE’s native meshing system are underway to generate meshes for fuel pin and hexagonal-array geometries. It is very likely that an input configuration could be created to perform a geometry-based parameter study using the new MOOSE mesh generation tools, when they become available.

8.5 Conclusions

The Stochastic Tools module in MOOSE presents exciting possibilities for improving both the sampling strategy and workflow of a hot channel factor survey. We have begun early testing on an input set for performing a parameter study on fissile content, which can be straightforwardly extended to other material composition-related uncertainties (coolant isotopics, for example). Although it is not possible to perform a geometry-based parameter study with the current workflow, pending additions to MOOSE (specifically, the new mesh generation tools in development under NEAMS) will likely make this type of study possible. Currently, we are limited to using the stochastic tools with the unaccelerated DFEM-SN or CMFD solvers due to the MultiApp-based implementation of NDA in Griffin. If additions were made in the future that enabled perturbation of parameters defined in nested sub-applications, it would be possible to use these tools while taking full advantage of the performance improvements brought by coarse mesh NDA. Although we do not yet have publishable early results using these tools, we are very optimistic about its future applications for hot channel factor determination.

9 Conclusions

Due to their ability to perform multiphysics coupling through the MOOSE framework, Griffin and NekRS were assessed individually for their ability to perform their respective physics calculations in an advanced hot channel factor simulation workflow. Since both Griffin and NekRS are under rapid development, their respective development teams are actively seeking user feedback to make the codes more user-friendly, performant, and accurate. Throughout the year, consistent communication was made between the authors of this report and the respective development teams which resulted in many of the identified code development gaps being addressed during the year.

The 3D heterogeneous pin-by-pin lead-cooled fast reactor assembly was modeled in Griffin to simulate the nominal condition which is needed for all HCF calculations. Griffin produced highly accurate results using all transport solvers, and significant gains in computational performance were observed over the course of this work. The DFEM-SN solver with the CMFD acceleration offered the lowest memory consumption and fastest wall clock time of all the solver options. However, memory consumption and instability of the "newton" type CMFD solver is still a barrier to using finer energy group structures, and currently we have only been able to run 9-group calculations. These 9-group cross sections were assessed for accuracy by comparison with MCNP reference solutions and larger energy group structure calculations. Furthermore, based on preliminary multiphysics studies, the HCF themselves do not appear to be very sensitive to differences in multigroup cross section sets. Several capabilities were added to Griffin this year due prioritization under this assessment including a new material time enabling memory and user-friendly microscopic cross sections, and easier post-processing of axial variables. Overall, Griffin is ready for the next stage of the project: performing coupled, perturbed simulations.

Multiple applications were employed to test the capabilities of NekRS for HCF calculation, such as mesh generation, mesh conversion, boundary assignment, averaging approach, conjugate heat transfer module, LES solver and RANS model. The results from Nek5000 were used as references for comparison. In summary, NekRS has the basic capabilities needed for HCF calculation. Some features in NekRS need to be improved to make it mature enough for HCF calculation for larger case.

Future work will involve coupling of NekRS and Griffin via the MOOSE MultiApp system and demonstration on a small problem which is computationally tractable for both solvers on the same machine. Griffin inputs will be re-generated using the newly developed MOOSE mesh tools for fast reactor geometry. MOOSE's Stochastic Tools modules will be explored for perturbing various conditions. Once NekRS is ready physics-wise and computationally, the use of MOOSE for in-memory coupling as well as stochastic perturbations will significantly streamline the workflow for performing hot channel factor simulations.

References

1. C. R. Stanek, Overview of DOE-NE NEAMS Program. United States: N. p., 2019. Web. doi:10.2172/1501761
2. M. DeHart et al. “Griffin User Manual,” INL/EXT-19-54247, April 2020.
3. Paul Fischer, Stefan Kerkemeier, Misun Min, Yu-Hsiang Lan, Malachi Phillips, Thilina Rathnayake, Elia Merzari, Ananias Tomboulides, Ali Karakus, Noel Chalmers, and Tim Warburton. 2021. NekRS, a GPU-Accelerated Spectral Element Navier-Stokes Solver. arXiv:2104.05829 [cs.PF]
4. E. R. Shemon, Y. Yu, and T. K. Kim, “Applications of SHARP Toolkit to SFR Challenging Problems: Evaluation of Hot Channel Factors and Demonstration of Zooming Capability,” ANL/NSE-18/4, Argonne National Laboratory, September 30, 2018.
5. Y. Yu, E. Shemon, T. K. Kim, “Hot channel factor evaluation for sodium-cooled fast reactors with multi-physics SHARP toolkit”, Proc. of NURETH 2019, Portland, OR, Aug 18-23, 2019.
6. E. R. Shemon, Y. Yu, and T. K. Kim, “High Fidelity Multiphysics Strategies with PROTEUS and Nek5000 for Fast Reactor Applications”, Proc. of M&C 2019, Portland, OR, Aug 25-20, 2019
7. Y. Yu, E. Shemon, T. K. Kim, and E. Merzari, “Evaluation of Hot Channel Factor for Sodium-Cooled Fast Reactors with Multi-Physics Tooolkit”, J. Nucl. Eng & Design. V 365, Aug 15, 2020, DOI: 10.1016/j.nucengdes.2020.110704
8. E. R. Shemon, Y. Yu, and T.K. Kim, “Demonstration of NEAMS Multiphysics Tools for Fast Reactor Applications”, ANL/NSE-20/25, Argonne National Laboratory, August 31, 2020.
9. Y. S. Jung and C. H. Lee, “Updates and Verifications of the PROTEUS System in FY19,” ANL/NSE-19/34, Argonne National Laboratory, September 30, 2019.
10. P. F. Fischer, J. W. Lottes, and Stefan G. Kerkemier, Nek5000 Web Page, http://nek5000.mcs.anl.gov, 2008.
11. J. Y. Ku, L. K. Chang and D. Mohr, “Analysis of IFR Driver Fuel Hot Channel Factors,” Fourth International Topical Meeting on Nuclear Thermal Hydraulics, Operation and Safety, Taipei, Taiwan, April 5-8, 1994.
12. C.J. Permann, et al., “MOOSE: Enabling massively parallel multiphysics simulation”, SoftwareX 11 (2020) 100430.
13. C. H. Lee and W. S. Yang, “MC2-3: Multigroup Cross Section Generation Code for Fast Reactor Anaysis,” ANL/NE-11-41 Rev. 1, Argonne National Laboratory, January, 2012.
14. G. Grasso, A. Levinsky, F. Franceschini, P. Ferroni, “A MOX-fuel core configuration for the Westinghouse Lead Fast Reactor,” ICAPP 2019 – International Congress on Advances in Nuclear Power Plants, France, 2019, May 12-15
15. Yoshihisa Ikusawa, Koji Maeda, Masato Kato, Masayoshi Uno. (2017) “Oxide-Metal Ratio Dependence of Central Void Formation of Mixed Oxide Fuel Irradiated in Fast Reactors,” Nuclear Technology 199:1, pages 83-95

16. MCNP Webpage, https://mcnp.lanl.gov/

17. Y. Wang, J. Ortensi, V. Laboure, J. Harter, O. Calvin, W. Dawn, "ISOXML – The Multigroup Cross Section Library Tools," (2020).

18. R. P. Hosteny, HMG4C - A Cross Section Homogenization Code Based on CCCC Data Sets, Argonne National Laboratory, 1978.

19. D. Medina, Okl: a unified language for parallel architectures, Ph.D. thesis, Rice University (2015).

20. D. S. Medina, A. St-Cyr, T. Warburton, OCCA: A unified approach to multi-threading languages, preprint arXiv:1403.0968

21. K. Swirydowicz, N. Chalmers, A. Karakus, T. War-burton, Acceleration of tensor-product operations for high-order finite element methods, Int. J. of High Performance Comput. App. 33 (4) (2019) 735–757.

22. A. Karakus, N. Chalmers, K. Swirydowicz, T. War- burton, A gpu accelerated discontinuous galerkin incompressible flow solver, J. Comp. Phys. 390 (2019) 380–404

23. Paul Fischer, Stefan Kerkemeier, Misun Min, Yu-Hsiang Lan, Malachi Phillips, Thilina Rathnayake, Elia Merzari, Ananias Tomboulides, Ali Karakus, Noel Chalmers, and Tim Warburton. 2021. NekRS, a GPU-Accelerated Spectral Element Navier-Stokes Solver. arXiv:2104.05829 [cs.PF]

24. Nek5000 Documentation Webpage, https://nek5000.github.io/NekDoc/

25. Paul K. Romano, Nicholas E. Horelik, Bryan R. Herman, Adam G. Nelson, Benoit Forget, and Kord Smith, "OpenMC: A State-of-the-Art Monte Carlo Code for Research and Development," *Ann. Nucl. Energy*, **82**, 90–97 (2015)

Nuclear Science and Engineering Division
Argonne National Laboratory
9700 South Cass Avenue, Bldg. 208
Argonne, IL 60439

www.anl.gov

Argonne National Laboratory is a U.S. Department of Energy
laboratory managed by UChicago Argonne, LLC